1963

A study of some aspects of linear programming and an approximate solution of a classical weight loading problem.

Byers, Austin L.

University of Kansas

A STUDY OF SOME ASPECTS OF LINEAR PROGRAMMING AND AN APPROXIMATE SOLUTION OF A CLASSICAL WEIGHT LOADING PROBLEM

AUSTIN L. BYERS

1963

A STUDY OF SOME ASPECTS OF LINEAR PROGRAMMING AND AN APPROXIMATE SOLUTION OF A CLASSICAL WEIGHT LOADING PROBLEM

by

Austin L. Byers
B. S., United States Naval Academy, 1954

Submitted to the Department of Chemical and Petroleum Engineering and the Faculty of the Graduate School of The University of Kansas in Partial Fulfillment of the Requirements for the Degree of Master of Science.

ACKNOWLEDGEMENTS

The author wishes to acknowledge his indebtedness to Dr. Charles F. Weinaug for his guidance in the graduate program in Petroleum Management.

The author is greatly indebted to Dr. Floyd W. Preston for his counsel, guidance and support in the preparation of this thesis. Dr. Preston has given, most willingly, of his time and talents to assist the author in his work on this thesis as well as in a multitude of other instances during the author's studies of Petroleum Engineering.

A special word of appreciation is due to Associate Professor Sherwood W. Newton who, as advisor in the School of Business, has assisted and encouraged the author in pursuing profitable studies in Business Administration.

Finally, the author is most grateful to the United States Navy for having granted him the time and opportunity for pursuing these studies in Petroleum Management.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Purpose

It would seem that the primary duty, and continuing responsibility of business managers (whether civilian or military), is to strive for the best possible economic results from the resources currently employed by, or available to them.

Linear programming has rapidly become one of the very important new techniques available for the management of resources of all kinds. The employment of digital computation equipment and appropriate digital computer codes enhances the scope of linear programming generally, and allows its efficacy to be used and explored more effectively.

The purpose of this thesis is two-fold; first, to demonstrate the application of linear programming to problems of the Navy Supply Corps and second, to present a new and original pragmatic approximation solution to a classical unsolved problem in an area related to linear programming.

In developing these topics for the thesis, it was deemed wise to include a rather full explanation of linear programming methodology. It is hoped that the work of this thesis will assist in promoting the practice of this

valuable management technique generally, but particularly within the U. S. Navy Supply Corps.

General

It seems paradoxical that World War II which caused so much waste, both in human and economic resources, could also cause the development of new theories and techniques designed to indicate the most efficient and economic utilization of resources. Historically, military necessity has, however, led not only to the development of new kinds of hardware, such as airplanes and digital computers, but also to the development of new scientific disciplines and conceptual understandings.

One such outgrowth of World War II is the new discipline known as "operations research" which has for its purpose optimizing the use of resources or, as stated by Jenny, "Operations research is concerned with optimizing the performance of a system. This requires the application of scientific methods, techniques, and tools."[1] Within this concept, linear programming is a technique or tool of operations research.

In the literature there are a great number of formalized definitions, and informal descriptions, which attempt

---

[1]Boulding, Kenneth E., W. Allen Spivey, et al., Linear Programming and the Theory of the Firm. New York: The Macmillan Co., 1960, p. 162.

to classify linear programming. Since simplified definitions of a complex subject are seldom of much real assistance in the understanding of that subject, most available definitions fail to adequately describe the full significance of linear programming. Among the several classifications to be found, linear programming has been variously described as ". . .one of the most important postwar developments in economic theory. . .,"[2] ". . .a branch of economic theory. . .,"[3] and ". . .a mathematical computing technique. . . ."[4] In a sense each description is correct but, at the same time, restricts the perspective in which linear programming should be viewed. For example, linear programming is used in economic theory and received its first great impetus there, yet it is also used in the physical sciences as well; linear programming is not just one mathematical computing technique but is, more properly, a mathematical approach (to problems) utilizing several computing techniques. Linear programming may be thought of as belonging to a larger class of problem solving approaches known as "mathematical programming."

---

[2]Dorfman, Robert, Paul A. Samuelson, and Robert M. Solow, Linear Programming and Economic Analysis. New York: McGraw-Hill Book Co., Inc., 1958, p. vii.

[3]Vajda, S., An Introduction to Linear Programming and the Theory of Games. London: Methuen & Co., Ltd., 1960, p. 11.

[4]Vajda, S., Readings in Linear Programming, New York: John Wiley and Sons, Inc., 1958, p. v.

Henderson and Schlaifer state that

> mathematical programming is not just an improved way of getting certain jobs done. It is in every sense a new way. It is new in the sense that double-entry bookkeeping was new in the Middle Ages, or that mechanization in the office was new earlier in the century, or that automation in the plant is new today.[5]

Programming problems, generally speaking, have to do with the optimum allocation of limited resources to satisfy specific objectives. Specifically, programming problems deal with situations where some resources, such as personnel, materials, equipment, and land are available, and are to be used in one or more ways to produce one or several results or products. There may be restrictions on the availability of one or more of the input resources; there may be restrictions on one or more of the expected output results or products; or there may be restrictions on both inputs and outputs. Within the restrictions, however, there are generally a vast number of possible or feasible solutions to the problem. From the great number of all possible solutions it is desired to find the one best or optimum solution, which will maximize or minimize (as required) some numerical quantity such as profit or cost. As Henderson and Schlaifer have said,

---

[5] Henderson, Alexander and Robert Schlaifer, "Mathematical Programming: Better Information for Better Decision Making", Harvard Business Review (May - June, 1954) p. 73.

> a group of limited resources must be shared among a number of competing demands, and all decisions are 'interlocking' because they all have to be made under that common set of fixed limits.[6]

It can be seen, then, that programming is intimately connected with decision making, in the management of resources.

As pointed out earlier, linear programming belongs to the larger classification of mathematical programming and deals with that class of programming problem for which all relations among the variables are linear. These relationships must be linear both in the constraints and in the objective function which is to be optimized. The linearity assumption is inherent in the discussion of linear programming and its solution techniques. This means, simply, that as one variable changes, other variables have some proportional rates of change. If the linearity assumption cannot realistically be made, or if the functions cannot be made linear by some suitable transformation of variables, then, linear programming cannot be applied.

Until a more rigorous mathematical statement may be made, let the following description, in a more-or-less abstract context, be a sufficient definition for discussion purposes: Linear programming is the mathematical theory of the minimization or maximization of a linear function

---

[6] *Ibid.*, p. 74.

subject to linear constraints; it deals with the interaction of many non-negative variables subject to certain restraining conditions.

Historical Perspective

Problems relating to maximization and minimization (extreme-value problems) have interested man ever since he first became aware of the concept. Euclid was concerned with finding the longest and the shortest straight lines that could be drawn from a point to the circumference of a circle. Dido solved one of the first (although non-linear) maximization problems by shreding the hide of an ox to bound the perifery of Carthage.

One of the most significant developments in forwarding the understanding and solution of maximization-minimization problems was the establishment of the branch of mathematics known as the "calculus of variations" which was first studied systematically by Euler and later by Lagrange.

Linear programming problems are of a type known as "constrained extreme-value problems" in which there is a stipulation that the variables must be non-negative (a value of zero being allowed). Classical methods of the calculus of variations may be used to solve constrained extreme-value problems with no stipulations regarding non-negative variables. Classical methods, however, are not applicable

when non-negative variables are required since they do not give any assurance that the values of those variables will not turn out to be negative.

Very few, if any, disciplines have ever evolved independently, and linear programming is no exception. A considerable portion of the mathematical theory of linear programming is drawn from the theory of linear inequalities and the theory of convex sets which were formulated during the past century.

An immediate forerunner to the development of linear programming was the work done in 1928 by the late John von Neumann, formerly professor of mathematics, Princeton University, when he formulated the mathematical theory of games ("Zur Theorie der Gesellshaftspiele," Mathematische Annalen, 1928, vol. 100, p. 295-300.)[7] Later, in 1944, this theory was expanded in collaboration with Oskar Morgenstern, Professor of Political Economy and Director, Economic Research Project, Princeton University, (Theory of Games and Economic Behavior, Princeton, New Jersey: Princeton University Press, 1944.)[8]

---

[7] Riley, Vera and Saul I. Gass, Linear Programming and Associated Techniques, Chevey Chase, Maryland: Operations Research Office, The Johns Hopkins University, 1958. p. 207.

[8] Ibid., p. 206.

In 1941 Frank L. Hitchcock, Professor Emeritus of Mathematics, Massachusetts Institute of Technology, presented the original statement and a solution of the so-called "transportation problem", ("The Distribution of a Product from Several Sources to Numerous Localities." Journal of Mathematics and Physics, Massachusetts Institute of Technology, August 1941, vol. 20, no. 3, pp. 224-230.)[9] The "transportation problem" (sometimes referred to as the "distribution problem") is one of several types of linear programming problems.

The second type of linear programming problem to be stated is what has been called the "diet problem" and was first presented in 1945 by George J. Stigler, Professor of Economics, Columbia University, ("The Cost of Subsistence," Journal of Farm Economics, May 1945, vol. 27, no. 2, pp. 303-314.)[10]

Each of these forerunners of the general linear programming model dealt with the optimization of a linear function subject to linear constraints but were not then specified to be linear programming problems.

The formulation of the general linear programming model was, however, primarily the work of George B. Dantzig, Marshall K. Wood and their associates in 1947, while

---

[9] Ibid., p. 284.

[10] Ibid., p. 400.

employed by the U. S. Air Force. At that time the U. S. Air Force was interested in developing a scientific basis for planning and budgeting decisions; the result was the organization of project SCOOP (Scientific Computation of Optimal Programs). Besides the immediate advantages accruing to the Air Force from the work of that research group, project SCOOP contributed the "key" to linear programming development and extension. The "key" was the development, by Dantzig, of a systematic procedure for solving the general linear programming problem; this procedure has been called the "simplex" method.

The simplex method of solving linear programming problems gave the necessary impetus to research in this field to allow linear programming to become a very important and valuable tool of modern theoretical and applied mathematics. Dantzig is quoted by Charnes and Cooper:

> The simplex procedure is a finite iterative method which deals with problems involving linear inqualities in a manner closely analogous to the solution of linear equations or matrix inversion by Gaussian elimination. . . . The term 'simplex' evolved from an early geometrical version in which (like in game theory) the variables were non-negative and summed to unity.[11]

Vajda, also quoted by Charnes and Cooper, states:

---

[11]Charnes, A. and W. W. Cooper, Management Models and Industrial Applications of Linear Programming, Vol. I, New York: John Wiley & Sons, Inc., 1961. p. 242.

> The name (simplex method) derives from the accident that one of the first examples to be tackled by this method contained the constraint
>
> $$\sum_i x_i = 1$$
>
> which is the equation of a simplex in n-dimensional geometry. The name (simplex method) is now used for the procedure whatever the form of the (linear) constraints.[12]

The simplex method, as originally devised by Dantzig, has certain drawbacks, such as being very time consuming and having the tendency to accumulate round-off errors during the computations. What has come to be known as the "revised simplex method", which is a streamlined version of the original procedure, was designed by Dantzig, Orchard-Hays and others at the RAND Corporation, in order to overcome the disadvantages of the original simplex method.

One other concept, that of duality, helped to promote the growth of linear programming by indicating how certain problems, otherwise intractable, could be solved using the theory of the Dual problem. It appears that Professor von Neumann first recognized that the dual problem existed, but it was Gale, Kuhn and Tucker who conducted the first investigations into duality and proved the first duality theorem.

---

[12] Ibid.

Contemporary Perspective

It became clear soon after the publication of the research report resulting from project SCOOP that the applications of linear programming far transcended only military application. Economists began, almost immediately, to apply linear programming to Leontief's input-output models of economic systems; mathematicians began to investigate the connections between the theory of games and linear programming; others have vigorously investigated various industrial applications of linear programming. Applications range from the original classical applications, to distribution and transportation systems, to the study of Kirchhoff laws of electrical-network theory,[13] and plastic-limit analysis in the design of structures.[14]

The rate of growth in the field of linear programming has been phenomenal and the theory is only in its infancy; today research is continuing at a tremendous rate, with no apparent end in sight.

Review of the Literature

The very large quantity of published literature in the field of linear programming is almost overwhelming;

---

[13] *Ibid.*, p. 628.

[14] *Ibid.*, p. 645.

for example, in 1957 Riley and Gass compiled a bibliography[15] containing well over one thousand references and since that time the available literature has increased considerably. Time has allowed for a review of only a small portion of the works available.

It has been noted that, as in most fields of study, inconsistencies among the authorities appear in the literature of linear programming. The most prevalent inconsistencies take the form of: (a) variations in the mathematical symbolism used, and (b) variations in the criteria used in solution methods. These inconsistencies are not serious and cause no real handicap; however, unless these differences are realized one who is unfamiliar with the literature may spend some uneasy moments in attempting to correlate certain aspects of linear programming among the various authorities.

In addition to the differences noted above, there is a wide range in the literature between the very theoretical approaches to linear programming and the "oversimplified" approaches; however, many authors attempt to arrive at some satisfactory degree of balance between the two extremes.

Although it is difficult to recommend any one particular reference as better than any other, perhaps the

---

[15]Riley. Op cit.

most concise and effective overall treatment, from both the theoretical and practical standpoints, are the books by Garvin,[16] and Charnes and Cooper.[17] The book by Charnes and Cooper is more comprehensive than the one by Garvin and it contains some of the more esoteric approaches to, and applications of, linear programming.

For anyone interested in a straight forward exposition of linear programming and some of its industrial applications with a minimum of theoretical appendages the book by Ferguson and Sargent[18] is recommended, although it is advised that the lack of theoretical explanation may ultimately become something of a disadvantage to the reader.

For those interested in a linear programming approach to economics the most celebrated and comprehensive work is that by Dorfman, Samuelson and Solow.[19] In this same category, but almost strictly on the theoretical level, is the book by Gale.[20]

---

[16] Garvin, Walter W., Introduction to Linear Programming, New York: McGraw-Hill Book Co., Inc., 1960.

[17] Charnes. Op. Cit.

[18] Ferguson, Robert O. and Lauren F. Sargent, Linear Programming: Fundamentals and Applications, New York: McGraw-Hill Book Co., Inc., 1958.

[19] Dorfman, Robert, Paul A. Samuelson and Robert M. Solow, Linear Programming and Economic Analysis, New York: McGraw-Hill Book Co., Inc., 1958.

[20] Gale, David, The Theory of Linear Economic Models, New York: McGraw-Hill Book Co., Inc., 1960.

Several, rather short, treatments of linear programming exist; probably the best are two books by Vajda.[21, 22] Greenwald[23] offers a short treatment of the simplex algorithm, as does Ficken.[24] The presentation given by Greenwald is somewhat more readible and less theoretical than the work by Ficken which is almost exclusively a theoretical exposition of the simplex algorithm.

The literature, represented in the periodicals, is even more extensive than that represented in book form, and no attempt has been made to survey more than a very small fraction of articles available.

Good, readable, general accounts of linear programming, including some of its applications may be found in the article by Henderson and Schlaifer;[25] the article

---

[21]Vajda, An Introduction. . .Games, Op. Cit.

[22]Vajda, Readings. . .Programming, Op. Cit.

[23]Greenwald, Dakota Ulrich, Linear Programming: An Explanation of the Simplex Algorithm, New York: The Ronald Press Co., 1957.

[24]Ficken, F. A., The Simplex Method of Linear Programming, New York: Holt, Rinehart and Winston, Inc., 1961.

[25]Henderson and Schlaifer, Op. Cit.

by Cooper and Charnes;[26] the article by Acrivos;[27] and the article by Rosander.[28] A good treatment of linear programming as it may be applied to long-range planning is found in the article by Rapoport and Drews.[29]

---

[26]Cooper, William W., and Abraham Charnes, "Linear Programming," Scientific American, August, 1954, vol. 191, no. 2. pp. 21-23.

[27]A. Acrivos, "Linear Programming: How Does it Work?", Chemical Engineering, 63:8, August, 1956, pp. 215-216.

[28]Rosander, A. C., "The Use of Linear Programming to Improve the Quality of Decisions," Industrial Quality Control, 12:9, March, 1956. pp. 11-16.

[29]Rapoport, Leo A. and William P. Drews, "Mathematical Approach to Long-Range Planning," Harvard Business Review, (May - June, 1962), pp. 75-87.

CHAPTER II

THE LINEAR PROGRAMMING PROBLEM

General

An impressive range of management problems have been solved by standard linear programming techniques. Also, a great deal of research has been successfully accomplished to determine simplified methods of solution and reduction (by approximations), of some previously intractable problems, to linear dimensions.

The apparent basic simplicity of linear programming theory together with the wide diversity of successful linear programming applications may be deceiving to those not fully aware of the complexities involved. Recognizing linear programming problems to be existent in raw data, and translating actual physical situations into the language and form suitable for operations with linear programming techniques is not a simple matter.

Recognizing Linear Programming Problems

Situations which come within the area suitable for treatment by linear programming methods may be grouped into three broad categories, although there is only one abstract statement of the linear programming problem, as will be presented below.

The categories of linear programming problems are:

Group I. This category includes situations in which the requirements for resources exceed the availability of the resources. Here, the problem is concerned with selecting the particular requirements which can use the available resources most effectively or economically.

Group II. This category pertains to situations having equal requirements for, and availability of resources. Problems of this type are solved if all demands are satisfied and all resources are used; however, allocation of the resources must be made in the most effective, or economical way when satisfying all the demands.

Group III. Included in this category are situations where the availability of resources exceeds the demand for the resources. Problems of this nature require that the most economical assignment of demands, to certain available resources, be made.

Regardless of any of the above three broad general categories into which a particular situation may be classified the following additional factors must also be evident in order to formulate a linear programming problem:

(a) There must be some objective which can be clearly stated and may be optimized and which can be expressed, in mathematical terms, as some linear function.

(b) There must be restrictions on the extent of attainment of the objective and these restrictions must be expressible, in mathematical terms, as a system of linear equations or linear inequalities.

(c) There must be alternatives among which to choose.

(d) The variables must be interrelated.

Once a problem has been determined to be a linear programming problem, in accordance with the above criteria, one may proceed to formulate the problem in mathematical language.

Mathematical Statement of the General Linear Programming Problem

It has been previously pointed out that linear programming deals with the interaction of many non-negative variables, which are subject to certain restraining conditions, for the purpose of determining maximum or minimum values. Some authorities state the standard linear programming problem as one of maximization (for example, see Saaty.[30]). Other authorities state the standard problem in terms of minimization (for example, see Churchman,

---

[30] Saaty, Thomas L., Mathematical Methods of Operations Research, New York: McGraw-Hill Book Co., Inc., 1959, pp. 167-168.

Ackoff, and Arnoff.[31]). The latter generalization will be made here; then, the mathematical statement of the general linear programming problem may be stated as follows:

Find the values of $X_1, X_2, X_3, \ldots, X_n$ which minimize:

$$\sum_{j=1}^{n} c_j X_j \equiv c_1X_1+c_2X_2+c_3X_3+\ldots+c_nX_n = Z \text{ (Min.)} \quad \ldots(2.1a)$$

subject to the conditions that:

$$X_j \geqq 0 \; (j = 1,2,3,\ldots,n) \quad \ldots(2.1b)$$

and

$$\sum_{j=1}^{n} a_{ij}X_j = b_i \quad (i = 1,2,3,\ldots,m) \quad \ldots(2.1c)$$

which may be written as:

$$a_{11}X_1 + a_{12}X_2 + a_{13}X_3 + \ldots + a_{1n}X_n = b_1$$

$$a_{21}X_1 + a_{22}X_2 + a_{23}X_3 + \ldots + a_{2n}X_n = b_2$$

$$a_{31}X_1 + a_{32}X_2 + a_{33}X_3 + \ldots + a_{3n}X_n = b_3$$

.

.

.

$$a_{m1}X_1 + a_{m2}X_2 + a_{m3}X_3 + \ldots + a_{mn}X_n = b_m$$

---

[31]Churchman, C. West, Russell L. Ackoff, E. Leonard Arnoff, *et al.*, *Introduction to Operations Research*, New York: John Wiley and Sons, Inc., 1957, pp. 326-327.

Or, given the column vectors:

$$P_j = \begin{bmatrix} a_{1j} \\ a_{2j} \\ a_{3j} \\ . \\ . \\ . \\ a_{mj} \end{bmatrix} \qquad (j = 1,2,3. \; . \; .,n). \; . \; .(2.1d)$$

$$P_o = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ . \\ . \\ . \\ b_m \end{bmatrix}$$

determine values of $X_1, X_2, X_3, \ldots, X_n$ which minimize the linear functional:

$$\sum_{j=1}^{n} c_j X_j = c_1X_1 + c_2X_2 + c_3X_3 + \ldots + c_nX_n = Z \text{ (Min)} \ldots (2.1a)$$

subject to the conditions that:

$$X_j \geqq 0 \ (j = 1,2,3,\ldots,n) \ldots (2.1b)$$

$$\sum_{j=1}^{n} P_j X_j = P_1X_1 + P_2X_2 + P_3X_3 + \ldots + P_nX_n = P_o \ldots (2.1e)$$

Where:

X_j are unknown variables, and $(i = 1,2,3,\ldots,m)$

a_{ij}, b_i, and c_j are given constants $(j = 1,2,3,\ldots,n)$

The problem is to select, out of the infinite number of solutions of equations (2.1c) or (2.1e) the set of solutions which contains only non-negative variables and for which the linear form of equation (2.1a) is a minimum. In other words the central problem is to minimize equation (2.1a) subject to equation (2.1b) and to the indeterminate system of equations (2.1c) or (2.1e). Equations (2.1c) are the constraints of the system, equation (2.1b) represents the non-negative conditions and equation (2.1a) is the objective function.

Thus the general linear programming problem has been stated in its standard (or canonical) form, where all the

constraints are equations, all the variables are required to be non-negative and the objective function is to be minimized.

Linear programming may also apply to situations where the constraints, instead of all being equations, are all inequalities or a mixture of equations and inequalities, or some variables may be negative, or the linear functional is to be maximized instead of minimized. Any of these alternate situations may be reduced to the standard form of linear programming problem as indicated above.

When a constraint is stated in the form of an inequality it may be changed into the form of an equation by adding a non-negative slack variable to the left hand side of the equation. For example, if we are given the less-than inequality:

$$a_{i1}X_1 + a_{i2}X_2 + a_{i3}X_3 + . \ . \ . + a_{in}X_n \leqq b_i$$

by adding a non-negative slack variable (S_i) to the left hand side of the inequation it is transformed into the following equation:

$$a_{i1}X_1 + a_{i2}X_2 + a_{i3}X_3 + . \ . \ . + a_{in}X_n + S_i = b_i$$

Similarily, if we are given a greater-than inequality:

$$a_{i1}X_1 + a_{i2}X_2 + a_{i3}X_3 + . \ . \ . + a_{in}X_n \geqq b_i$$

the inequation may be transformed into an equation by adding a non-negative slack vector (S_i) to the right hand side of the inequality (or, transposed to the left hand side it becomes [$-S_i$]):

$$a_{i1}X_1 + a_{i2}X_2 + a_{i3}X_3 + \cdot\ \cdot\ \cdot + a_{in}X_n = b_i + S_i$$

$$a_{i1}X_1 + a_{i2}X_2 + a_{i3}X_3 + \cdot\ \cdot\ \cdot + a_{in}X_n - S_i = b_i$$

If some variable is not restricted to non-negativity (i.e., an unrestricted variable) it may be transformed into a non-negative constraint to suit the standard linear programming form. Since any number may be expressed as the difference of two non-negative numbers the variable can be defined in terms of two new variables:

$$X_j = X_j' - X_j'' \quad . \ . \ . \quad (2.2)$$

where: $X_j' \geqq 0$ and $X_j'' \geqq 0$

Substituting the above relationship (2.2) into equation (2.1c) or (2.1e) the system again conforms to the standard form. Hadley[32] offers an interesting demonstration that the value of X_j uniquely determines the values of X_j' and X_j''.

If, instead of wishing to minimize, it is desired to maximize an objective function so that:

---

[32]Hadley, G., Linear Programming, Reading, Mass: Addison-Wesley Publishing Co., Inc., 1962. p. 169.

$$\sum_{j=1}^{n} c_jX_j = c_1X_1 + c_2X_2 + c_3X_3 + \ldots + c_nX_n = Z \text{ (Max)}$$

Then, this relationship may be rewritten:

$$-\sum_{j=1}^{n} c_jX_j = -c_1X_1 - c_2X_2 - C_3X_3 - \ldots - c_nX_n = Z \text{ (Min)}$$

Vajda states:

> The problem of maximizing an objective function is not fundamentally different from that of minimizing one, because maximizing is equivalent to minimizing the negative.[33]

Thus, the maximization problem is transformed into a minimization problem of the standard form. This procedure, however, is not identical to the duality concept discussed below.

The Mathematical Model

The concept of a mathematical model is intimately associated with the subject of problem formulation. In the preceding section we have described a mathematical model of the general linear programming problem in its most abstract sense.

A mathematical model is a device which aspires to be a replica of a system which is to be studied in some detail; without constructing a mathematical model one cannot

---

[33] Vajda, Introduction. . .Programming, Op. Cit., p. 12.

formulate the linear programming problem. Mathematical models may vary in the degree of faithfulness with which they adhere to an actual physical situation, depending upon the desired accuracy which is expected or the purpose for which the model is intended.

Once a mathematical model is established it may be studied from various perspectives, utilizing various assumptions in order to gain insight into the "real life" circumstances of the problem. The versatility of a mathematical model will be demonstrated in Chapter V and Chapter VI.

The Primal Problem

It is convenient, when discussing linear programming problems, to speak of the "primal" problem and its "dual" problem.

The standard problem statement as indicated above is generally considered the statement for the primal problem, although it is really immaterial which problem statement is called the primal and which is called the dual.

To facilitate discussion of the primal problem and the concepts presented thus far, some simple (bivariable) examples will be used for illustration. Examples with only two variables will be used, obviously, in order

to allow for graphical representation, since bi-variable relationships may be presented on cartesian coordinates.

(Example Problem No. 1)

Assume that a U. S. Navy Supply Officer finds that he must order a quantity of gasoline for stock and for immediate use. He has available storage capacity for 1,600 gallons. He normally issues two grades of gasoline in about equal quantities (each grade of gasoline has approximately equivalent demand rates). The supply officer knows that he has an immediate requirement for at least 1,000 gallons of either kind, or a combination of both kinds of gasoline, but due to special operational requirements all of one type of gasoline (X_1) plus one-half of the other type (X_2) must equal or exceed 800 gallons. The officer has been informed that there are only 700 gallons of type X_2 and 800 gallons of type X_1 available to fill the order at the present time. The gasoline costs 15¢ per gallon for type X_1 and 13¢ per gallon for type X_2. The supply officer must determine what quantities of each type of fuel to obtain in order to minimize the cost to the government.

In order to formulate the problem in terms suitable for a linear programming approach, a mathematical model has to be constructed, as follows:

The objective which is desired is the minimization of cost, or: $0.15X_1 + 0.13X_2 = Z$ (Min) . . . (2.3a) which conforms to equation (2.1a) of the standard linear programming problem.

Since negative quantities of gasoline would not be a reasonable expectation, we may state:

$$X_1 \geqq 0$$
$$X_2 \geqq 0 \quad . . . (2.3b)$$

These relationships (2.3b) correspond to equation (2.1b) of the general linear programming problem and also indicate that a graphical representation of the problem in cartesian coordinates will be applicable in only the first quadrant.

The conditions or constraints of the problem, corresponding to the system of equations represented by relationship (2.1c) of the standard linear programming problem, may be stated mathematically as follows:

$X_1 + X_2 \leqq 1{,}600$ (Storage Capacity) . . . . . . (2.3c)

$X_1 + X_2 \geqq 1{,}000$ (Order Quantity) . . . . (2.3d)

$X_1 + \frac{1}{2}X_2 \geqq 800$ (Special Usage Requirement). . (2.3e)

$X_1 \leqq 800$ (Limitation of X_1 Availability) (2.3f)

$X_2 \leqq 700$ (Limitation of X_2 Availability) (2.3g)

Utilizing figures (2.1a) through (2.1e) we may demonstrate the meaning of the constraining inequations; in figure (2.1f) the meaning of the objective function will be shown.

Equation (2.3c) describes the storage limitations in terms of a less-than inequality. This condition is indicated by the line in figure (2.1a) at the maximum limit (extreme); the arrow shows that this constraint may take on an infinite number of other positions toward the origin of the graph.

Equation (2.3d) describes the minimum order quantity and is expressed as a greater-than inequality. This relationship is shown by the line in figure (2.1b) at its minimum limit (extreme), with the arrow indicating that this constraint may take on an infinite number of positions away from the origin of the graph.

Similarly equations (2.3e), (2.3f) and (2.3g) are shown by lines at the extreme limits of the constraints in figures (2.1c), (2.1d) and (2.1e) respectively. Again, arrows indicate the direction in which the constraint may move to assume an infinite number of other positions.

Figure (2.1f) shows the objective function at several of the possible positions out of the infinite number of other positions which it could assume. Since the objective

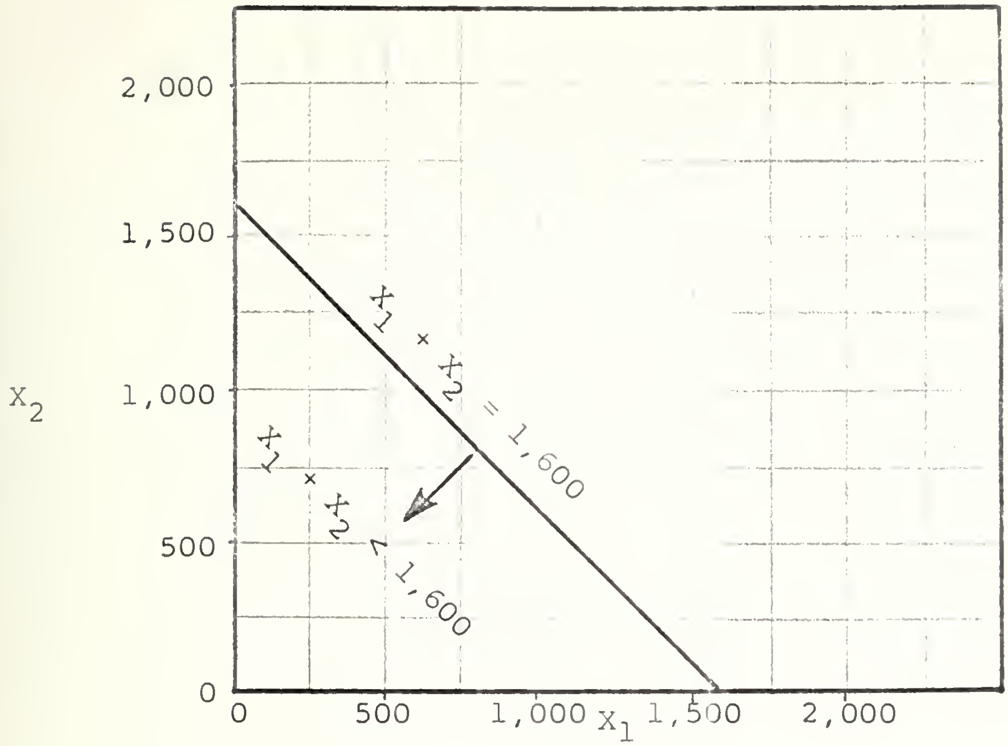


FIGURE 2.1a

GRAPHIC REPRESENTATION OF EQUATION (2.3c)

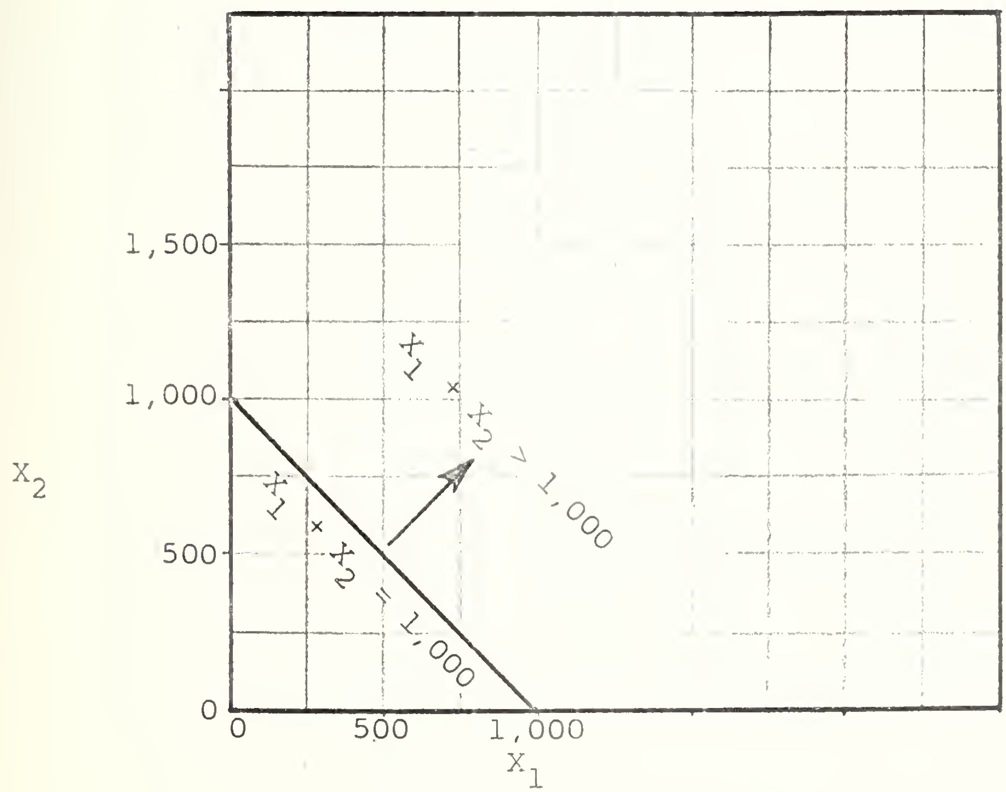


FIGURE 2.1b

GRAPHICAL REPRESENTATION OF EQUATION (2.3d)

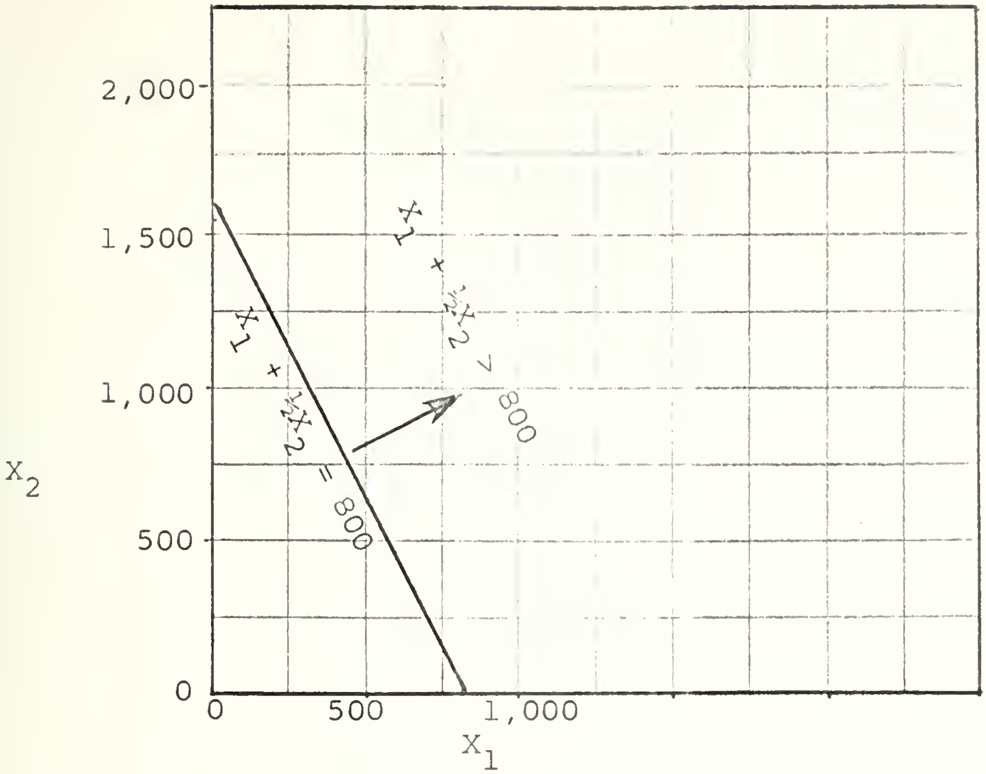


FIGURE 2.1c

GRAPHICAL REPRESENTATION OF EQUATION (2.3e)

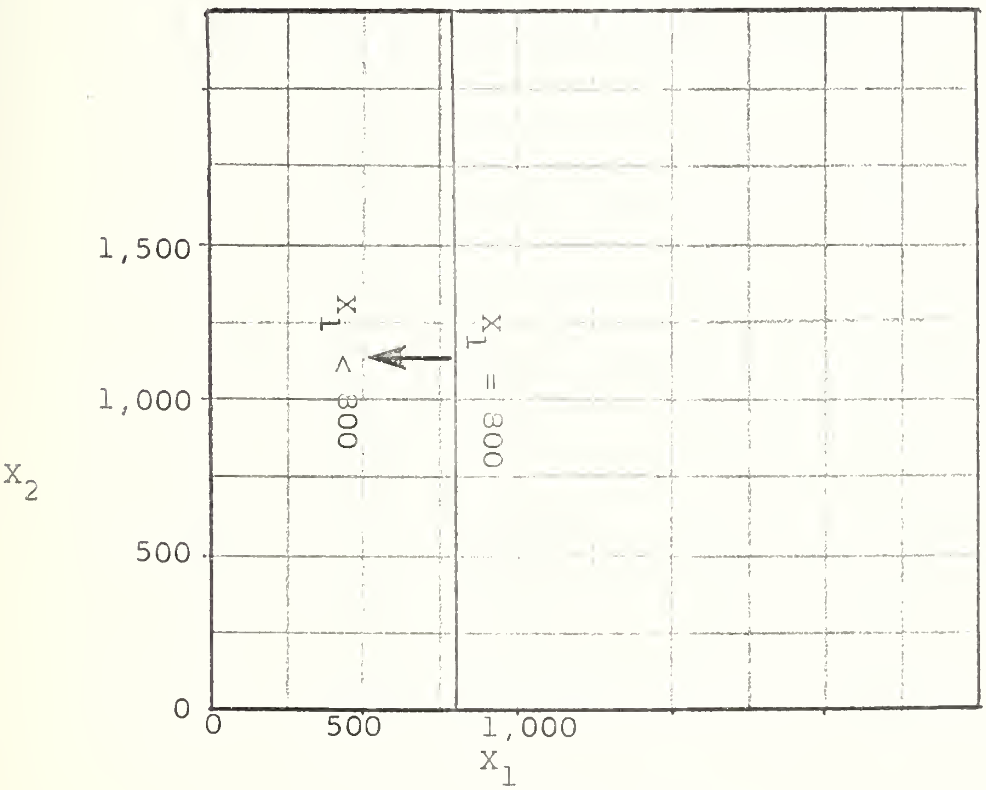


FIGURE 2.1d

GRAPHICAL REPRESENTATION OF EQUATION (2.3f)

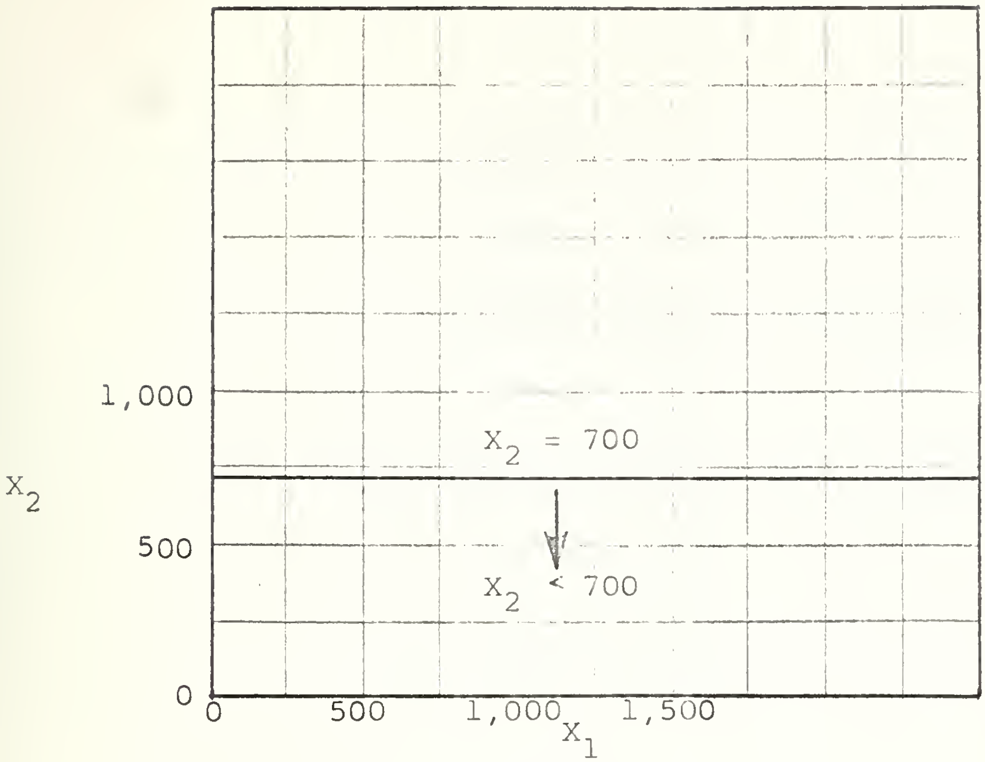


FIGURE 2.1e

GRAPHICAL REPRESENTATION OF EQUATION (2.3g)

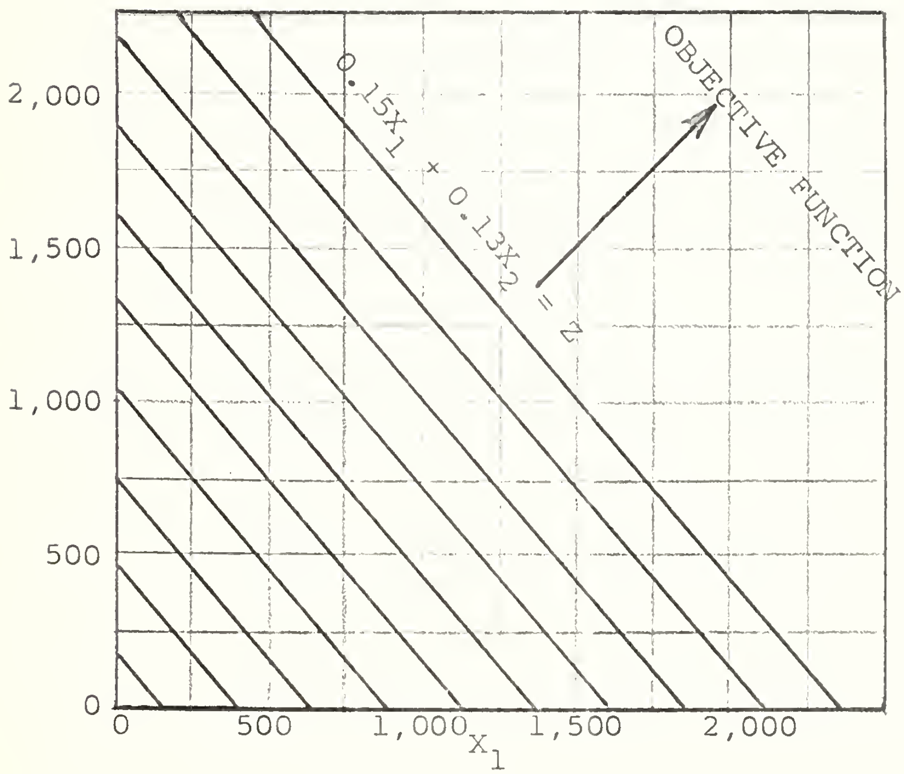


FIGURE 2.1f

GRAPHICAL REPRESENTATION OF THE OBJECTIVE FUNCTION EQUATION (2.3a)

function is not constrained (except by the non-negativity of the variables) it may take on any position from zero, at the origin, out to infinity.

Superimposing all the individual graphs one above the other we can see a consolidated representation of the problem, as a whole, in figures (2.2a) and 2.2b). The enclosed polygon in figure (2.2a) indicates the area of feasible solutions (shaded area); any combination of X_1 and X_2, within the shaded area, would satisfy the constraining conditions defined in equations (2.3c) through (2.3g). However, since we are seeking an optimum solution, the solution will occur at one of the extreme points of the convex polygon formed by the constraining boundaries.

Since the problem is to minimize the objective function, imagine the lines (with a slope of -15/13) as shown in figure (2.1f), progressing from the origin outward in figure (2.2b). The first point of tangency of this line with the polygon, shown in figure (2.2b) by the shaded area, is at the point $X_1 = 600$, $X_2 = 400$. This point, then, is the maximum order quantity for minimum expenditure. The cost of this order may easily be computed:

$$(\$0.15)(600) + (\$0.13)(400) = \$142.00$$

To make certain that this actually is the minimum cost we may examine the cost of the other extreme points. If the

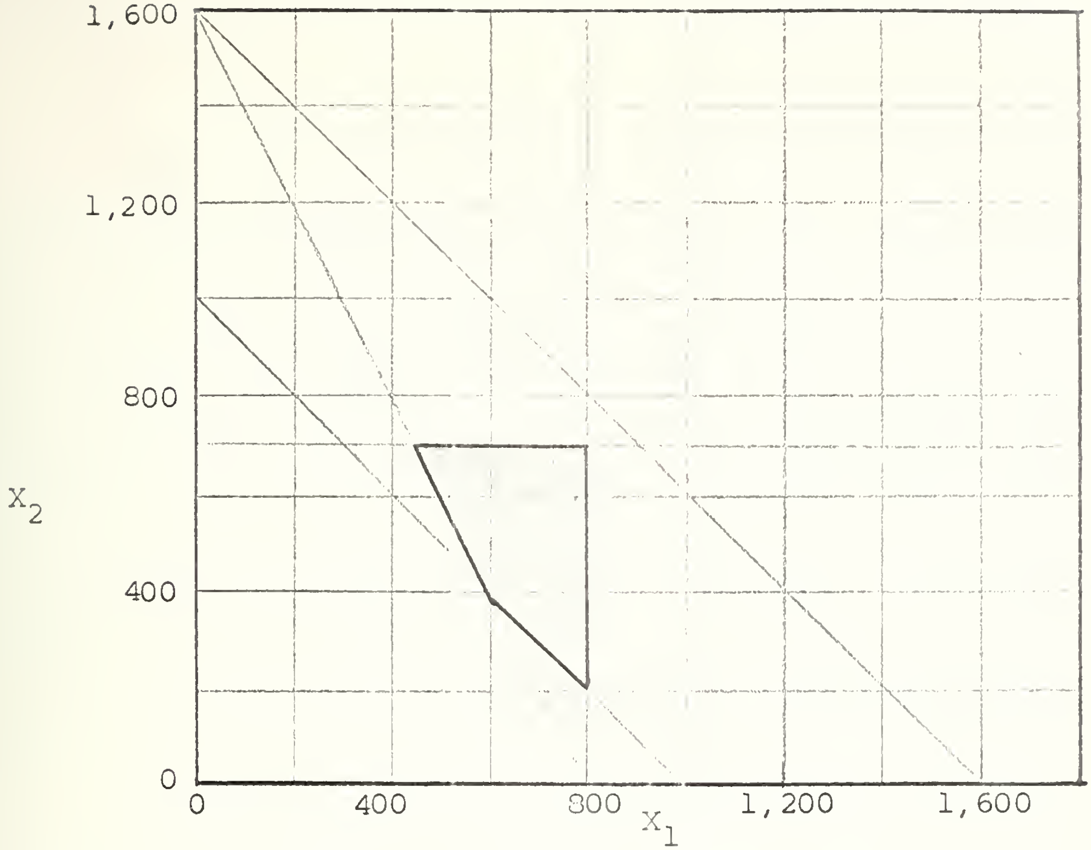


FIGURE 2.2a

GRAPHICAL REPRESENTATION OF THE CONSTRAINING EQUATIONS OF EXAMPLE PROBLEM NO. 1

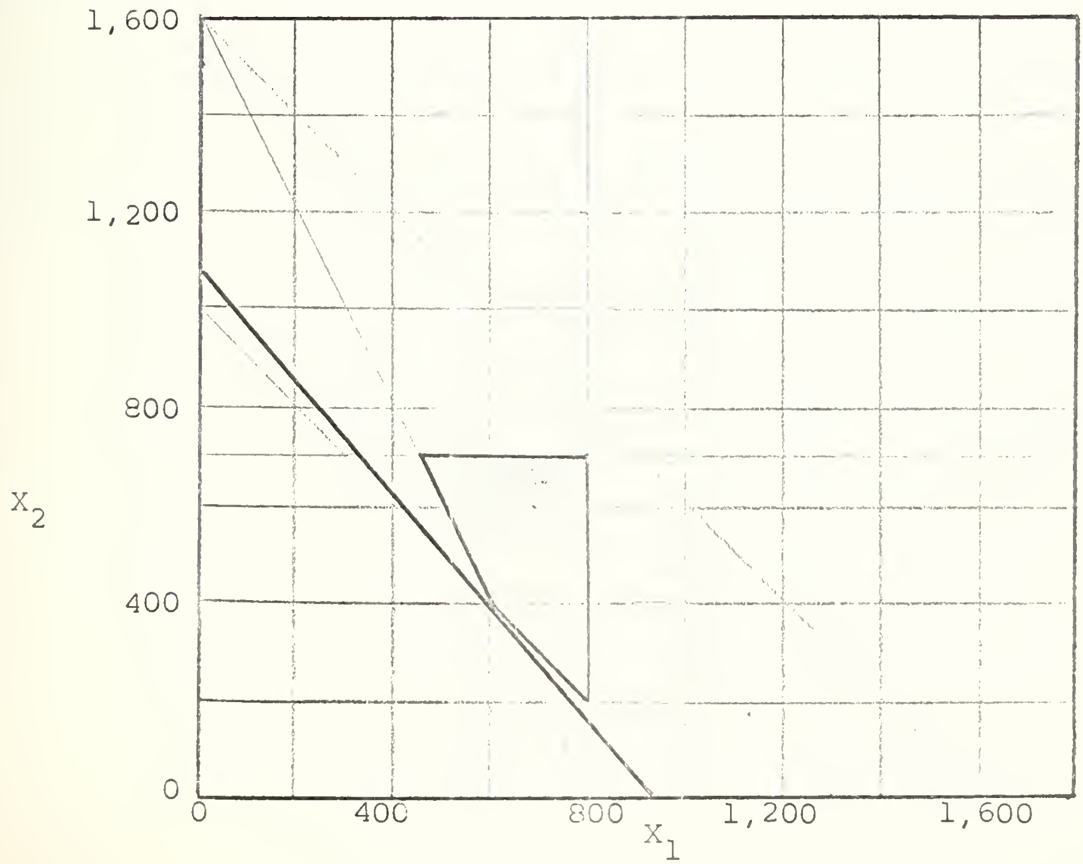


FIGURE 2.2b

order were $X_1 = 450$, $X_2 = 700$ the cost would be \$158.50. If the order were $X_1 = 800$, $X_2 = 200$ the cost would be \$146.00. If it was decided to order the maximum of available quantities $X_1 = 800$, $X_2 = 700$, the cost would be \$211.00. Therefore, by means of the method first used it is seen that we obtained, directly, the minimum cost of the order which was the objective originally sought.

(Example Problem No. 1 [Modification No. 1])

Continuing with the example used above, assume that the supply officer had instituted a policy that there may be an imbalance of stock in favor of larger quantities of X_2, since it is somewhat more difficult to obtain and it is less expensive than X_1. The officer, however, has specified that X_2 should not exceed X_1 by more than 200 gallons. The new conditions, required by the policy, may be shown as follows:

$X_2 - X_1 \geqq 0$ (Minimum imbalance of stock). .(2.3h)

$-X_1 + X_2 \geqq 0$

$X_2 - X_1 \leqq 200$ (Maximum imbalance of stock). .(2.3i)

$-X_1 + X_2 \leqq 200$

These additional constraints, along with the original constraining conditions are shown in figure (2.3). It can be seen that a new polygon of feasible solutions is formed. Proceeding as before, imagine the same line, representing the objective function ($0.15X_1 + 0.13X_2 = Z$) to be progressing from the origin outward. The first point of tangency is now: $X_1 = 533.33$; $X_2 = 533.33$. The new minimum cost of the order, under the constraints, is \$149.33.

One would imagine that the supply officer, by instituting such a policy, as described above, was attempting to maintain efficient economical operations. It is readily apparent, though, that he has been successful only in equalizing his order for the two types of gasoline while increasing the cost of the total order.

This latter example points out one of the many, but probably least obvious, advantages of linear programming: the evaluation of policy. When certain criteria are considered essential to operations these criteria can be evaluated in terms of their additional cost, as shown above. Perhaps, when it is shown exactly what certain policies cost, it may be determined that their "essentiality" is only imaginary.

Linear programming has found many useful applications in private enterprise. Here, again, as in the example given

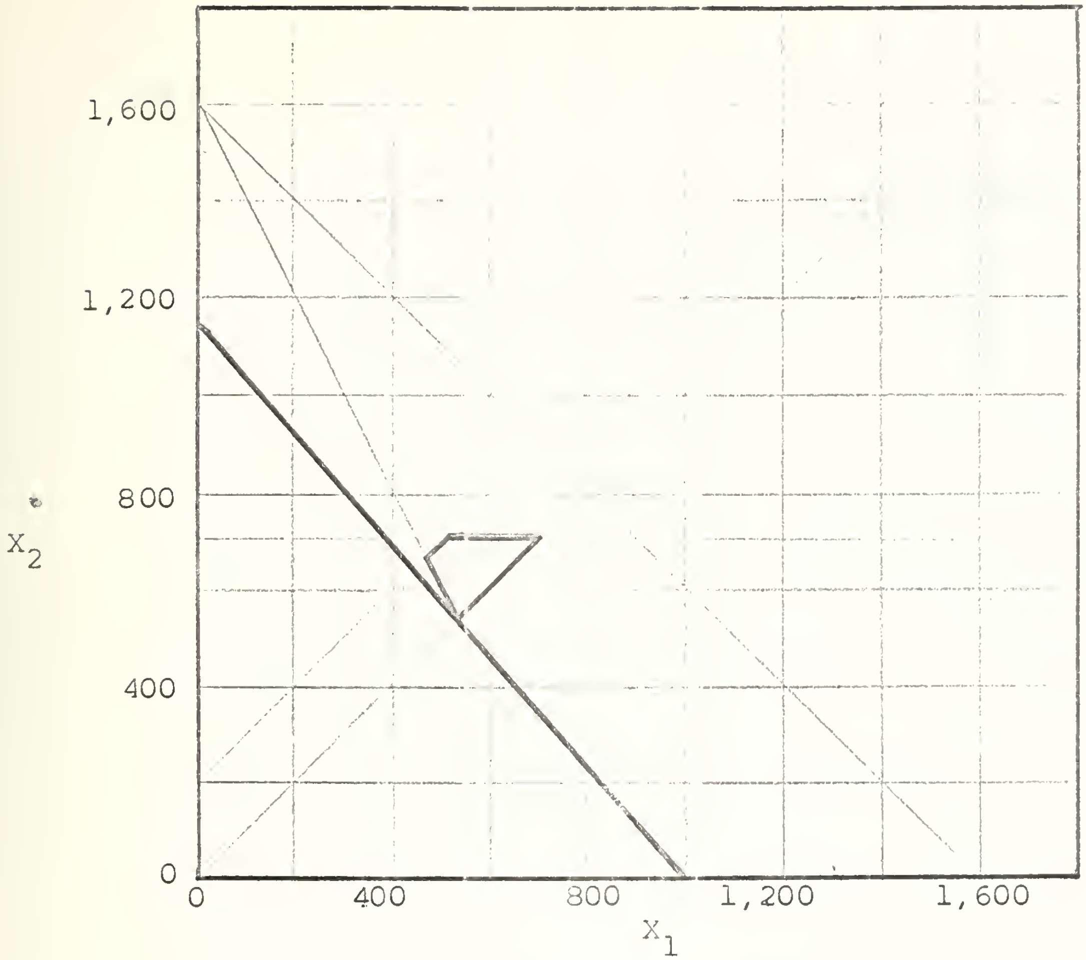


FIGURE 2.3

GRAPHICAL SOLUTION OF EXAMPLE PROBLEM NO. 1
(MODIFICATION NO. 1)

above, one may be seeking to minimize costs; however, unlike Navy or governmental operations generally, which are unconcerned with profit "per se", one most probably would be more concerned in private industry with maximizing profits of the business.

(Example Problem No. 2)

Consider the following very simple example: an oil company manufactures two types of lubricating oil from one kind of crude oil stock. Each gallon of product requires equal quantities of one gallon of the crude oil, but the crude oil is available only to the extent of 12,000 gallons per month and it costs 5¢ per gallon. In order to make the two products a certain machine processing operation is required where the cost of operation is $2.00 per hour. Product X_1 requires one-half of an hour of machine time operation per each gallon produced; product X_2 requires one hour of machine time to produce one gallon of product. The machines are available for production of these two products for only 500 hours of each month. If the company can make a profit of 5½¢ per gallon on product X_1 and 7½¢ per gallon on product X_2 what amounts of these two products should be produced in order for the company to receive the maximum profit?

First, the problem may be stated mathematically in consistent units (i.e., monetary units in this case):

crude oil available: (12,000)(5¢) = $600.00

machine time available: (500)($2.00) = $1,000.00

X_1 machine requirement coefficient: (½)($2.00)=$1.00

X_2 machine requirement coefficient: (1)($2.00)=$2.00

The objective function for maximum profit is:

$0.055X_1 + 0.075X_2 = X$ (MAX) . . . .(2.4a)

Negative quantities of the products cannot be made; therefore:

$$X_1 \geqq 0 \qquad X_2 \geqq 0 \qquad \text{. . . . .(2.4b)}$$

The constraining conditions are:

$X_1 + X_2 \leqq 600$ (crude oil limitation) . . . .(2.4c)

$X_1 + 2X_2 \leqq 1{,}000$ (machine processing limitation(2.4d)

The convex polygon of all feasible solutions is shown by the shaded area in figure (2.4) which graphically represents the above relationships. Since the objective in this case is the maximization of profits we must imagine that the line representing the objective function is progressing from infinity toward the origin (at a slope of -0.055/0.075). The first point of tangency to the convex polygon, which is the maximum point, is at $X_1 = 200$; $X_2 = 400$. The maximum profit is, therefore, $41.00.

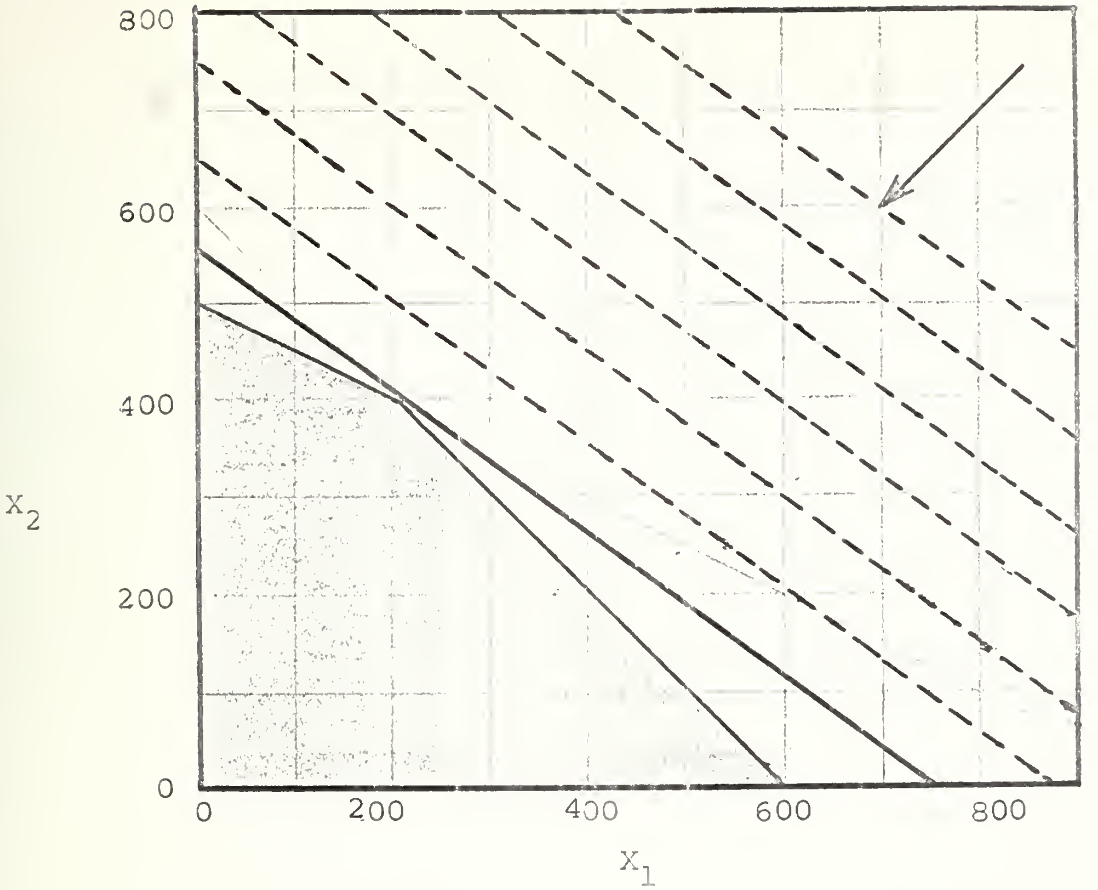


FIGURE 2.4

GRAPHICAL SOLUTION OF EXAMPLE PROBLEM NO. 2

It is interesting to note how the various profit figures affect the objective function and how it, in turn, determines the optimum quantities of production. Figures (2.5a) and (2.5b) reproduce the convex polygon formed by the constraining relationships of the previous example and with various new objective functions having changed profit margins. In figure (2.5a) the solid line represents an objective function of equal profitability between the products X_1 and X_2 (for example, assume: $0.060X_1 + 0.060X_2 = Z$). The dashed line of figure (2.5a)

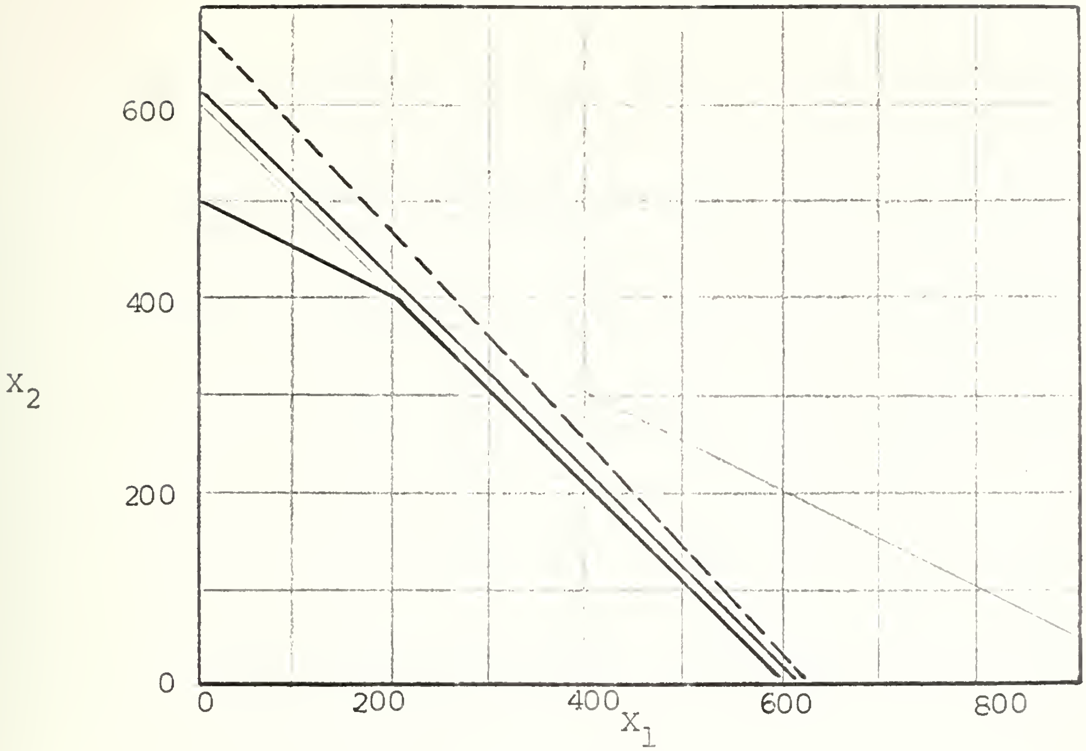


FIGURE 2.5a

PROFIT MARGIN VARIATION (EQUAL PROFITABILITY)

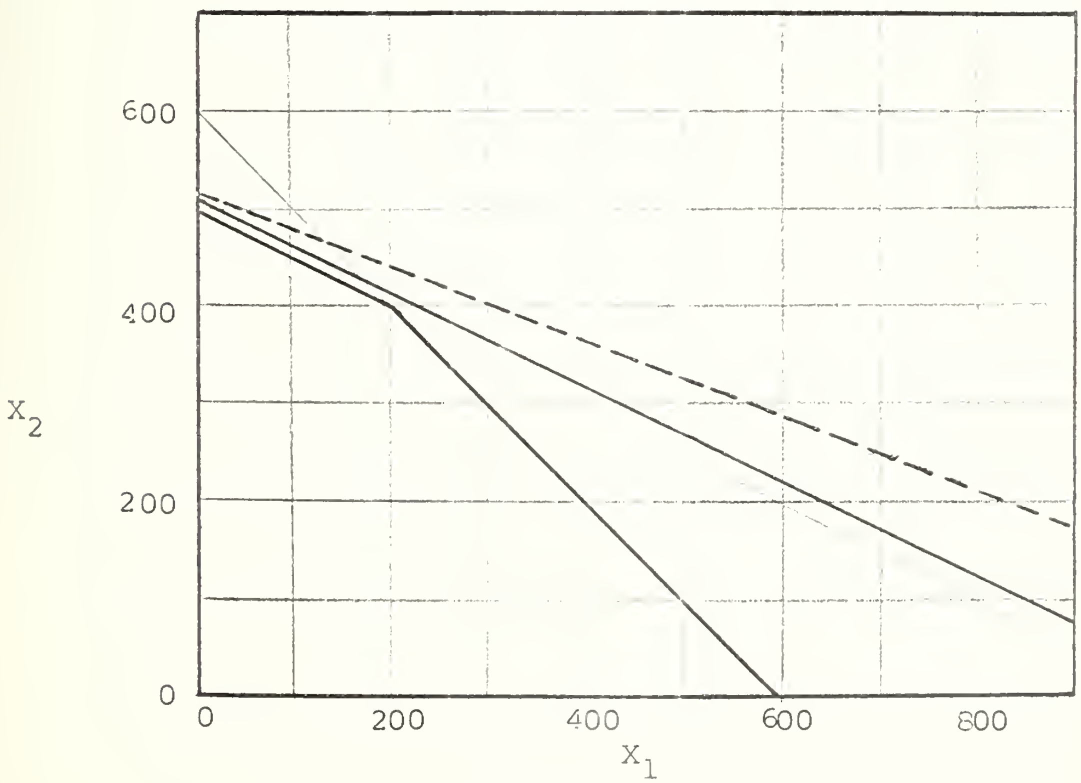


FIGURE 2.5b

PROFIT MARGIN VARIATION (UNEQUAL PROFITABILITY)

represents a profitability of X_1 slightly exceeding X_2 (for example, assume: $0.0605X_1 + 0.060X_2 = Z$). Note that in the case of equal profitability the line representing the objective function would come to rest directly upon the line representing the constraining relationship describing the limit of availability of the crude oil ($X_1 + X_2 = 600$). This situation indicates that any feasible solution on that line (including the maximum solution of the previous example) would be equally profitable. Possible* solutions in this instance are:

$X_1 = 200;\ X_2 = 400;\quad Z = \$36.00$

$X_1 = 300;\ X_2 = 300;\quad Z = \$36.00$

$X_1 = 400;\ X_2 = 200;\quad Z = \$36.00$

$X_1 = 500;\ X_2 = 100;\quad Z = \$36.00$

$X_1 = 600;\ X_2 = 0;\quad Z = \$36.00$

Determining such a fact as this might have important bearing on management decisions since all elements necessary for final effective decisions may not enter into any basis of the problem solution because of certain non-quantifiable factors concerning the product data. For example, it might be more desirable, for good labor relations, to produce one

---

*Note that intermediate solutions such as: $X_1 = 249.1$, $X_2 = 349.9$, are equally valid if non-integral values are acceptable.

agreeable product rather than some other product which may be offensive to personnel for some reason; or, one product might be less hazardous to make than another product. Knowing that equivalent profitability could be gotten from various product mixes, management would be able to make more valuable decisions. Under certain circumstances, however, as in this example, when a slight fluctuation in the profit margin occurs (like the incremental change shown by the dashed line of figure [2.5a]) there is only one optimum product mix.

Figure (2.5b) represents substantially the same idea of equal profitability but with regard to a profit margin differential between the products. The solid line represents the relationship of product X_2 being twice as profitable as product X_1 (for example, assume an objective function: $0.05X_1 + 0.10X_2 = Z$). Here the objective function would come to rest upon the line representing the constraining relationship resulting from limited machine capacity. Again, we have a situation where we may have an infinite number of solutions (assuming that the solution may contain non-integer results) anywhere from $X_1 = 0$; $X_2 = 500$ up to and including $X_1 = 200$; $X_2 = 400$; in any case the profit return would be a constant $50.00.

Next, if we allow a small increment in the profit margin of product X_2 (assume, now, an objective function: $0.05X_1 + 0.101X_2 = Z$). In this case the dashed line, representing the objective function, first touches the polygon of feasible solutions at $X_1 = 0$; $X_2 = 500$ indicating that maximum profitability can be made from production of only product X_2 (with a profit return of \$50.10).

In the same manner, had product X_1 increased slightly in profitability instead of product X_2 (assume an objective function of: $0.0501X_1 + 0.10X_2 = Z$). The maximum profit then would be \$50.02 and occur only with the production of $X_1 = 200$; $X_2 = 400$; and this was the original solution obtained.

In addition to showing how various product combinations may, under certain circumstances, return identical profits the foregoing illustrations indicate how some slight variation in profit margin may become very critical to the decision making process. This points out that seldom will a solution to a given problem become static. Continual vigilence and review of pertinent conditions is obviously necessary to insure that optimum solutions, once gotten, remain optimal.

The Dual Problem

To every linear programming problem involving maximization there exists a specific corresponding minimization problem involving the same data and having the same optimal solution. The first problem explained above, is called the primal; its counterpart is known as the dual.

As mentioned previously, it is really immaterial which problem is considered the primal and which problem is considered the dual. It is equally valid that for every linear programming problem requiring minimization there also exists a specific maximization problem involving the same data and having the same optimal solution.

Utilizing the data of the previous example of profit maximization, considered in the last section, the pertinent data is displayed in table (2.1). It can be seen from table (2.1) that the primal problem is formulated by taking the data row by row whereas the dual problem is formulated by taking the data column by column.

TABLE 2.1

THE PRIMAL/DUAL PROBLEM

	PRIMAL / DUAL	MAXIMIZE X_1	X_2	
MINIMIZE	W_1	$a_{11} = 1$	$a_{12} = 1$	$600 \leqq b_1$
	W_2	$a_{21} = 1$	$a_{22} = 2$	$1{,}000 \leqq b_2$
		$0.055 \geqq c_1$	$0.075 \geqq c_2$	SOLUTION \$41.00

PRIMAL PROBLEM

$$c_1 X_1 + c_2X_2 = Z \quad \text{(maximum)}$$

$$a_{11} X_1 + a_{12}X_2 \leqq b_1$$

$$a_{21} X_1 + a_{22}X_2 \leqq b_2$$

DUAL PROBLEM

$$b_1W_1 + b_2W_2 = Z \quad \text{(minimum)}$$

$$a_{11}W_1 + a_{12}W_2 \geqq c_1$$

$$a_{21}W_1 + a_{22}W_2 \geqq c_2$$

Specifically, the primal and dual problems, for the given example, are formulated as follows:

PRIMAL PROBLEM

$$0.055X_1 + 0.075X_2 = Z \text{ (Max)}$$

$$X_1 + X_2 \leqq 600.00$$

$$X_1 + 2X_2 \leqq 1{,}000.00$$

DUAL PROBLEM

$$600W_1 + 1{,}000W_2 = Z \text{ (Min)}$$

$$W_1 + W_2 \geqq 0.055$$

$$W_1 + 2W_2 \geqq 0.075$$

Graphically, the primal problem was presented in figure (2.4). The graphical representation of the dual problem is shown in figure (2.6). It can be seen in figure (2.6) that the polygon of feasible solutions extends from the lines representing the constraining relationships out toward infinity.

In the dual problem the objective being sought is to find the minimum value and, as before, imagine the line representing the objective function to proceed from the origin (with a slope of -600/1,000 = -3/5) toward infinity. In figure (2.6) it can be seen that the first point of tangency of the line with the polygon of feasible solutions is at $W_1 = 0.035$; $W_2 = 0.02$. The value of the objective function at this point is: (600)(0.035) + (1,000)(0.02) = \$41.00 which is the same solution reached in the maximization of the primal problem.

The proof of the duality theorem, which is somewhat involved, is given by Gale.[34]

Comment on Problem Formulation Context

It should be pointed out, at this point, if it is not already obvious, that the context of a particular problem is really irrelevent (except in the matter of final

---

[34]Gale. Op. Cit., pp. 78-82.

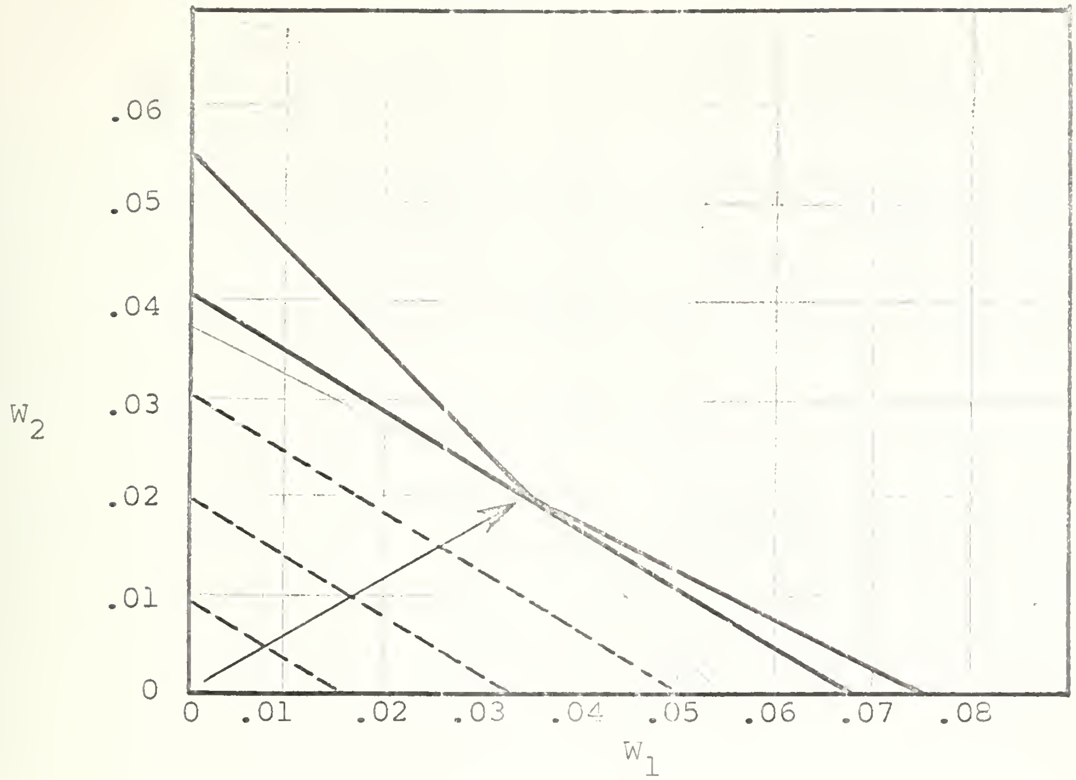


FIGURE 2.6

GRAPHICAL SOLUTION OF THE DUAL OF EXAMPLE PROBLEM NO. 2

interpretation of the results of the solution). What is important and indispensable is the proper construction of the mathematical model in the beginning of problem formulation. When a problem is stripped of its verbiage and a mathematical model is properly constructed in terms of interrelated variables (say: $X_1, X_2, \ldots, X_n$) it does not matter what the variables really stand for (assuming they have been properly defined at the outset). For example, in the foregoing problem, where profit from the manufacture of two products was maximized, the problem could have

been stated, just as well, as follows: A U. S. Navy Supply Officer has a disbursing clerk who can either compute pay or audit vouchers, or do both jobs sequentially. The clerk can process one pay record (on the average) in 3.30 minutes; to audit one voucher it takes the clerk 4½ minutes (on the average). Both operations require the use of a typewriter at the rate of one minute per document for both pay records and vouchers; there are 10 hours of typewriter time available to the clerk during one week. Each operation also requires the use of a desk calculator at the respective rates of one minute per pay record and 2 minutes per voucher; a total of 16 2/3 hours of time are available to the clerk for using the calculator during one week. The supply officer desires that the disbursing clerk produce as much work as possible during the week, therefore, the officer must determine how to best assign the work to the clerk.

In essence the problem just described is identical to the preceding problem although the context is entirely different; by constructing the mathematical model this becomes evident. Let:

X_1 = pay records processed

X_2 = vouchers processed

Since it is desired to maximize the clerk's weekly output, then the objective function is:

$$3.3/60\ X_1 + 4.5/60\ X_2 = Z\ (\text{Max})$$

subject to the restrictions:

$$1/60\ X_1 + 1/60\ X_2 \leqq 10$$

$$1/60\ X_1 + 2/60\ X_2 \leqq 16\ 2/3$$

Or,

$$0.055X_1 + 0.075X_2 = Z\ (\text{Max})$$

$$X_1 + X_2 \leqq 600$$

$$X_1 + 2X_2 \leqq 1{,}000$$

and

$$X_1 \geqq 0$$

$$X_2 \geqq 0$$

Thus, we have the same problem as before but in an entirely different setting. In this latter problem, of course, the solution is exactly the same as before ($X_1 = 300$; $X_2 = 400$; $Z = 41$). The interpretation, from the model to the physical situation is the only difference; the numerical data and the mathematical model are identical.

CHAPTER III

BASIC METHODS OF PROBLEM SOLUTION EMPLOYED IN LINEAR PROGRAMMING

General

Until the development of the simplex method, there were no general solution techniques available with which to solve linear programming problems. Ferguson and Sargent report that "the simplex method is the basic linear programming method from which the other known methods have been derived."[35] Gale states that the simplex method is ". . .one of the most celebrated computational procedures in recent mathematical history.", and ". . .so successful has the simplex method been that from the literature one might be led to believe that linear programming *is* the simplex method."[36]

Many modifications and extensions have been developed since the original simplex method was first reported. A "revised simplex" procedure has been established and is the algorithm generally used in computer codes, but Hadley

---

[35] Ferguson and Sargent. *Op. Cit.*, p. 71.

[36] Gale. *Op. Cit.*, p. 105.

reports that "the revised simplex method has not met with wide acceptance for hand computations."[37]

Another outgrowth of the simplex method has been the development of the "transportation method" (or, sometimes called the "distribution method"), and the "modified distribution method" (or, "MODI method").

The use of any of the linear programming solution methods requires the manipulation of a great deal of arithmetic, and the real power and usefulness of linear programming lies in the solution of very large mathematical models where the employment of hand computations would be unrealistic, and only digital computation machinery can adequately perform the necessary arithmetic. There is a place, of course, for hand computations, when relatively small problems are being considered, and electronic computation equipment is not available.

In the following sections of this chapter a discussion of the simplex method and the modified distribution (MODI) method will be presented.

Simplex Method (Minimization)

The literature abounds in examples of using the simplex method to solve maximization linear programming problems. There are not so many examples of the use of

---

[37] Hadley. Op. Cit., p. 215.

this method to solve minimization problems, however. The method of solution is essentially the same whether one is considering maximization or minimization problems, but there are slight differences which are not altogether obvious.

In this discussion a minimization problem will be treated by converting it to a maximization problem and proceeding with the simplex computations. Where there are differences in proceedure between minimization and maximization, the differences will be pointed out in the ensuing discussion.

The discussion of solution methods may be facilitated by the use of an example; in this discussion example problem No. 1 (modification No. 1), presented and solved graphically in chapter II, will be used. The mathematical model (equations [2.3a] through [2.3i]), introduced in chapter II, is reproduced below:

Objective Function: $$0.15X_1 + 0.13X_2 = Z \text{ (Min)}$$

Constraints:

$$\begin{aligned}
X_1 + X_2 &\leqq 1{,}600 \\
X_1 + X_2 &\geqq 1{,}000 \\
X_1 + \tfrac{1}{2}X_2 &\geqq 800 \\
X_1 &\leqq 800 \\
X_2 &\leqq 700 \\
-X_1 + X_2 &\geqq 0 \\
-X_1 + X_2 &\leqq 200 \\
X_j &\geqq 0
\end{aligned}$$

Prior to attempting a solution of a system of linear inequalities (as shown above) the inequations must be converted to equations as indicated in chapter II. This conversion may be accomplished by the introduction of slack variables. These variables accomplish just what their name implies: slack variables take up (or, are assigned the quantities of) any difference, or slack between the total quantity, called for by the equation, and the total quantity assigned to the primary variables.

An inequality of the "less-than" ($\leqq$) form may be converted to an equation by the addition of a slack variable to the left hand side of the inequation. An inequality of the "greater-than" ($\geqq$) form may be converted to an equation by the addition of a slack variable to the right hand side of the inequation; then, this slack variable is subtracted when transposed to the left hand side of the inequation.

Since the slack variables merely absorb unused resources, they add nothing to the profit and incur nothing in the way of costs. Therefore, slack variables must carry zero (0) profit and zero (0) cost coefficients; however, these variables may be assigned penalty coefficients under certain circumstances.

The mathematical model, shown above, consists of the following system, after the inclusion of slack variables:

Objective Function:

$$0.15X_1 + 0.13X_2 + 0X_3 + 0X_4 + 0X_5 + 0X_6 + 0X_7 + 0X_8 + 0X_9 = Z\ (\text{Min})$$

$$\begin{aligned}
X_1 + X_2 + X_3 &= 1{,}600 \\
X_1 + X_2 - X_4 &= 1{,}000 \\
X_1 + \tfrac{1}{2}X_2 - X_5 &= 800 \\
X_1 + X_6 &= 800 \\
X_2 + X_7 &= 700 \\
- X_1 + X_2 - X_8 &= 0 \\
- X_1 + X_2 + X_9 &= 200
\end{aligned}$$

In all solution methods employed in linear programming one always proceeds toward a final optimal solution by starting with a feasible solution (which is not necessarily optimal but is a possible solution to the system of equations). Generally, in the simplex method, a first feasible solution is gotten by allowing the structural variables (X_1 and X_2 in the present example) to equal zero. This is equivalent to saying that nothing will be produced and no resources will be utilized.

Now, notice in this example that if we take the approach of allowing X_1 and X_2 to equal zero we cannot

obtain a feasible solution. This misfortune comes about because we would be saying, among other things, that:

$$X_4 = -\ 1{,}000$$

$$X_5 = -\ \ 800$$

and we have, therefore, violated the non-negative variable requirements ($X_j \geqq 0$).

To overcome this difficulty we may adopt a device solely for the purpose of obtaining a first feasible solution. The device which will be used is the introduction of "artificial variables" which have no physical relation to our problem and, therefore, must not enter (at a positive level) into the basis of the final optimal solution. To prevent the artificial variables from becoming a part of the final optimal solution we will enter them at an arbitrarily high cost (M) since we are attempting minimization of the objective function. Since the artificial variables have no bearing on the problem and must not enter the final optimal solution the high cost (M) may be carried throughout the computations without being actually specified numerically, if it is always remembered that it (M) is the largest number in the system.

The mathematical model, after introducing the artificial variables, consists of the following system:

Objective Function:

$$.15X_1+.13X_2+0X_3+0X_4+0X_5+0X_6+0X_7+0X_8+0X_9+MX_{10}+MX_{11}+MX_{12}=Z \text{ (Min)}$$

$$\begin{array}{lllllllllllllll}
 & X_1+ & X_2+ & X_3 & & & & & & & & & & = & 1{,}600 \\
 & X_1+ & X_2 & & -X_4 & & & & & & +X_{10} & & & = & 1{,}000 \\
 & X_1+ & \tfrac{1}{2}X_2 & & & -X_5 & & & & & & +X_{11} & & = & 800 \\
 & X_1 & & & & & +X_6 & & & & & & & = & 800 \\
 & & X_2 & & & & & +X_7 & & & & & & = & 700 \\
- & X_1+ & X_2 & & & & & & -X_8 & & & & +X_{12} & = & 0 \\
- & X_1+ & X_2 & & & & & & & +X_9 & & & & = & 200
\end{array}$$

At this point we encounter the first difference between minimization and maximization problems. Above we have entered the cost of the artificial variables as a positive high cost (M). In a maximization problem we would have to enter the artificial variables at an arbitrary high negative cost (-M) in order to prevent their entry into the final optimal solution.

In chapter II it was stated that "maximizing is equivalent to minimizing the negative," Hadley offers a proof of this relationship.[38] Numerically this concept may be demonstrated as follows:

---

[38] Ibid., pp. 130-131.

given the following series:

5, 4, 3, 2, 1, 0, -1, -2

then five (5) is the maximum. Negating the series we have:

-5, -4, -3, -2, -1, 0, +1, +2

then, minus five (-5) is the minimum.

Using this fact, as demonstrated above, we restate the objective function as follows:

$$-.15X_1-.13X_2-OX_3-OX_4-OX_5-OX_6-OX_7-OX_8-OX_9-MX_{10}-MX_{11}-MX_{12}=Z(\text{Max})$$

The tableau format is the universal device used for linear programming computations and the present system of equations is shown in tableau (3.1a); it is a representation of the first feasible solution (with $X_1=X_2=X_4=X_5=X_8=0$). Column (c_b) contains the cost coefficients (c_j) of the respective variables in the current basis (these variables are found in the column headed: "vectors in basis"). The main body of the tableau contains the detached coefficients (a_{ij}) of the variables (where no entry is made, a zero is to be understood). As a convenience all the cost coefficients of the objective function have been entered above the tableau; and the columns headed "Key Row" and "Ratio", as well as the row headed "Key Column" have been added for discussion purposes.

Column (b), Row (z_j) represents $\Sigma\ c_b b$; in the first tableau this is computed as follows:

$$
\begin{aligned}
(0)(1{,}600) &= 0 \\
(-M)(1{,}000) &= -1{,}000M \\
(-M)(\ \ 800) &= -\ \ 800M \\
(0)(\ \ 800) &= 0 \\
(0)(\ \ 700) &= 0 \\
(-M)(\ \ \ \ 0) &= 0 \\
(0)(\ \ 200) &= \underline{0\qquad\quad} \\
\text{Total} &\quad -1{,}800M
\end{aligned}
$$

Other values in row (z_j), for the various columns (X_j) are computed in the same manner as above (i.e.,: $\Sigma\ c_b a_{ij}$).

Row ($z_j - c_j$) is computed by taking the (z_j) as computed above and subtracting the respective column vector cost (c_j); for example, $z_1 = -M$, $c_1 = -.15$, then, $z_1 - c_1 = -M - (-.15) = -M + .15$. Normally row ($z_j$) is not recorded since our main concern will be with row ($z_j - c_j$).

After an initial tableau has been constructed one is ready to begin iterations, in a step-by-step procedure, toward a final optimal solution.

<u>Step 1</u>. To begin an iteration find the most negative ($z_j - c_j$); identify its column (X_j) as the "key column" (note the double asterisk in the "key column" row of the tableaux). The X_j will be the variable entering the basis in the new tableau.

<u>Step 2</u>. Divide each number in the "key column" into its corresponding value in column (b). (Note "ratio" column.)

Find the quotient having the smallest positive value; identify that row (X_i), associated with the smallest quotient, as the "key row". (Note the double asterisk in the "key row" column of the tableaux.) The variable (X_i) is the variable which will be eliminated from the basis.

In attempting to determine the smallest positive valued quotient it may happen that there are "ties" where two quotients have the same smallest positive value. In the case of ties the following procedure will allow one to continue the general procedure:

a. If the "ties" are not caused by zeros in the numerators of the ratios then the ratio which has the largest denominator is chosen and that row is identified as the "key row". One would then go on to step 3.

b. If the "ties" are caused by zeros in the numerator of the ratio, then the denominators of the ratios are each divided into each value in their respective rows starting at the constant column and moving to the right comparing at each column. When one of the new ratios becomes smaller than the other, the tie is considered broken and the row containing the smaller value (i.e., smaller ratio) is designated the "key row" and one then would proceed to step 3.

Step 3. Identify the number at the intersection of the "key row" and the "key column" as the "key number". (Note the circled numbers in the tableaux.)

Step 4. Divide all numbers in the "key row" by the "key number" starting with column (b) and going through column (X_n). These quotients are entered into a new tableau in the corresponding row, and that row, under the column heading "vectors in basis", is designated by the X_j of the "key column" of the previous tableau; this is the entering variable. The cost of the entering variable is placed under column (c_b) in its respective row. The remainder of rows under columns (c_b) and "vectors in basis" are reproduced in the new tableau from the previous tableau.

Step 5. The values for the remaining boxes (or cells) of the new tableau (including rows [z_j] and [$z_j - c_j$] as well as column [b]) are computed as follows:

$$\begin{matrix}\text{value in}\\ \text{new}\\ \text{tableau}\end{matrix} = \begin{bmatrix}\text{value in}\\ \text{old}\\ \text{tableau}\end{bmatrix} - \frac{\begin{matrix}\text{corresponding}\\ \text{"key row"}\\ \text{number}\end{matrix} \times \begin{matrix}\text{corresponding}\\ \text{"key column"}\\ \text{number}\end{matrix}}{(\text{"key number"})}$$

As an example, column X_8 of the second tableau (i.e., tableau 3.2b) is computed below:

ROW	VALUE IN OLD TABLEAU	−	CORRESPONDING "KEY ROW" NUMBER x CORRESPONDING "KEY COLUMN" NUMBER / "KEY NUMBER"	=	VALUE IN NEW TABLEAU
X_3	0	−	$\frac{(-1)(1)}{1}$	=	1
X_{10}	0	−	$\frac{(-1)(1)}{1}$	=	1
X_{11}	0	−	$\frac{(-1)(\frac{1}{2})}{1}$	=	½
X_6	0	−	$\frac{(-1)(0)}{1}$	=	0
X_7	0	−	$\frac{(-1)(1)}{1}$	=	1

Old row: X_{12}

New row: X_2

This was the "key row";

$$\text{new value} = \frac{\text{old value}}{\text{"key number"}} = \frac{-1}{1} = -1$$

X_9	0	−	$\frac{(-1)(1)}{1}$	=	1

Step 6. After the new tableau is completed inspect row $(z_j - c_j)$ to determine if there are any negative values (i.e., $z_j - c_j < 0$); if all are positive or zero $(z_j - c_j \geqq 0)$ an optimal solution has been reached and no further iterations are necessary. If any $z_j - c_j < 0$ an optimum solution has not been reached and further iterations are required; one begins again at step 1, going through the entire procedure (steps 1 through 6). If, however, some

$z_j - c_j < 0$ and all $a_{ij} < 0$ then the solution is unbounded or infinite (further iterations would be of no benefit since an optimum solution does not exist).

Once an optimum has been reached the solution is found for the vectors in the basis (X_i) in their respective row under column (b). "Shadow prices" of variables not in the basis may be found under the respective columns (X_j) in row ($z_j - c_j$). Shadow prices represent the solution of the dual problem. As mentioned earlier, it should be noted that differences exist among the various authorities relative to certain aspects of handling the simplex computations. Paramount among the differences is that some authors do not negate the objective function initially when performing a minimization operation. This results in a change of criteria for selecting the variables to enter the basis at each iteration as well as a change of criteria for determining an optimum final solution; for example, Greenwald follows this method.[39] Consistency is an all important concern in using the simplex method, and these variations are of no great difficulty if one is consistent; however, one should be aware of procedural differences existing in the literature.

---

[39] Greenwald. Op. Cit., pp. 63-66.

					|←— SLACK VARIABLES						—→| ←ARTIFICIAL VARIABLES→				
		c_j	-.15	-.13	0	0	0	0	0	0	0	-M	-M	-M		
c_b	VECTORS IN BASIS	b	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	KEY ROW	RATIO
0	X_3	1,600.	1	1	1											1,600/1
-M	X_{10}	1,000.	1	1		-1						1				1,000/1
-M	X_{11}	800.	1	½			-1						1			800/½
0	X_6	800.	1					1								800/0
0	X_7	700.		1					1							700/1
-M	X_{12}	0.	-1	1						-1				1	**	0/1
0	X_9	200.	-1	1							1					200/1
z_j		-1,800M	-M	-2½M	0	M	M	0	0	M	0	-M	-M	-M		
$z_j - c_j$		-1,800M	-M +.15	-2½M +.13	0	M	M	0	0	M	0	0	0	0		
KEY COLUMN				**												

KEY NUMBER 1

TABLEAU 3.1a

					SLACK VARIABLES						ARTIFICIAL VARIABLES				
		c_j	-.15	-.13	0	0	0	0	0	0	0	-M	-M	-M		
c_b	VECTORS IN BASIS	b	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	KEY ROW	RATIO
0	X_3	1,600.	2	0	1					1				-1		1,600/2
-M	X_{10}	1,000.	2	0		-1				1		1		-1	**	1,000/2
-M	X_{11}	800.	1½	0			-1			½			1	-½		800/1½
0	X_6	800.	1	0				1		0				0		800/1
0	X_7	700.	1	0					1	1				-1		700/1
-.13	X_2	0.	-1	1						-1				1		0/-1
0	X_9	200	0	0						1	1			-1		200/0
z_j		-1,800M	-3½M	0	0	M	M	0	0	-1½M	0	-M	-M	1½M		
$z_j - c_j$		-1,800M	-3½M +.28	0	0	M	M	0	0	-1½M +.13	0	0	0	2½M -.13		
KEY COLUMN			**													

KEY NUMBER 2

TABLEAU 3.1b

					<— SLACK VARIABLES					—>	<— ARTIFICIAL VARIABLES		—>		
		c_j	-.15	-.13	0	0	0	0	0	0	0	-M	-M	-M		
c_b	VECTORS IN BASIS	b	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	KEY ROW	RATIO
0	X_3	600.	0	0	1	1				0		-1		0		600/1
-.15	X_1	500.	1	0		-½				½		½		-½		500/-½
- M	X_{11}	50.	0	0		3/4	-1			-¼		-3/4	1	¼	**	50/(3/4)
0	X_6	300.	0	0		½		1		-½		-½		½		300/½
0	X_7	200.	0	0		½			1	½		-½		-½		200/½
-.13	X_2	500.	0	1		-½				-½		½		½		500/-½
0	X_9	200.	0	0		0				1	1	0		-1		200/0
z_j		-50M	0	0	0	-3/4 M	M	0	0	¼ M	0	3/4 M	-M	-¼ M		
$z_j - c_j$		- 50M -140.00	0	0	0	-3/4M +.14	M	0	0	¼M -.01	0	7/4M -0.14	0	3/4M +.01		
KEY COLUMN						**										

KEY NUMBER 3/4

TABLEAU 3.1c

					SLACK VARIABLES						ARTIFICIAL VARIABLES					
		c_j	-.15	-.13	0	0	0	0	0	0	0	-M	-M	-M		
c_b	VECTORS IN BASIS	b	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	KEY ROW	RATIO
0	X_3	533.333	0	0	1	0	4/3			1/3		0	-4/3	-1/3		
-.15	X_1	533.333	1	0		0	-2/3			1/3		0	2/3	-1/3		
0	X_4	66.667	0	0		1	-4/3			-1/3		-1	4/3	1/3		
0	X_6	266.667	0	0		0	2/3	1		-1/3		0	-2/3	1/3		
0	X_7	166.667	0	0		0	2/3		1	2/3		0	-2/3	-2/3		
-.13	X_2	533.333	0	1		0	-2/3			-2/3		0	2/3	2/3		
0	X_9	200.00	0	0		0	0			1	1	0	0	-1		
z_j		0	0	0	0	0	0	0	0	0	0	0	0	0		
$z_j - c_j$		-149.333	0	0	0	0	.1866	0	0	.0366	0	M	M -.1866	M -.0366		
KEY COLUMN																

TABLEAU 3.1d

FINAL (OPTIMUM) TABLEAU

The Modified Distribution (MODI) Method

The modified distribution method is a somewhat less involved method of solution than is the simplex method. The MODI method, being applicable to only certain types of problems (generally called "transportation type" problems), is not a universal method of solution as is the simplex computational procedure.

The transportation type problems to which the MODI method may be applied take the general form, as follows:

Objective Function: $$\sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} X_{ij} = Z \text{ (Min)}$$

Constraints: $$\sum_{j=1}^{n} X_{ij} = a_i \quad i = 1,2,3,\ldots,m$$

$$\sum_{i=1}^{m} X_{ij} = b_j \quad j = 1,2,3,\ldots,n$$

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$$

$$X_{ij} \geqq 0 \quad \text{(all } i,j\text{)}$$

These problems may be identified in actual situations when a particular system is in balance (i.e., when available resources exactly equal the requirements or demands for the resources), and the resource to be allocated among the various demands is homogeneous. In addition, the resources and demands must be expressible in a single unit

(such as gallons, pounds, hours, etc.). Realizing that we are still discussing linear programming with its inherent linearity requirement, it might appear that these additional limitations would severly restrict the usefulness of the MODI method; actually this is not the case. Transportation type problems occur very frequently and the MODI method is often applied with great effectiveness.

As before, we shall use an example to demonstrate how the transportation type problem may be formulated and how the MODI method may be used for its solution.

(Transportation Problem)

Assume that a U. S. Navy Supply Officer has requisitions from five destroyers, each ordering various quantities of a particular lubricating oil. These demands are as follows:

DESTROYER	1	2	3	4	5
DEMAND QUANTITY (GALLONS)	100	60	40	75	25

The supply officer has three storage locations where he keeps the lubricating oil and he knows that to get the oil to the various piers where the ships are tied up it will cost the government money as shown in the following transportation cost table:

COST (¢/gal.) OF TRANSPORTING LUBRICATING OIL FROM STORAGE LOCATION TO DESTINATION (DESTROYER)

DESTROYER	1	2	3	4	5
STORAGE LOCATION					
1	3	2	3	4	1
2	4	1	2	4	2
3	1	0	5	3	2

The supply officer knows that he will have no trouble filling the requisitions because he has exactly sufficient lubricating oil on hand to meet all of the demands, as follows:

STORAGE LOCATION	QUANTITY (gal.) AVAILABLE
1	100
2	125
3	75

The problem facing the supply officer, then, is to get the desired quantities of lubricating oil to each destroyer at the least cost to the government.

The tableau format is the generally accepted computational data display device used in the MODI method as it was in the simplex method; however, the MODI tableau is not the same as the simplex tableau.

By examining the first tableau (3.2a) it can be seen that:

a. The quantities of lubricating oil available at the storage locations are entered under column (S);

b. The requirements of the various destroyers are entered in row (D);

c. Transportation costs associated with shipping from a storage location to a particular destroyer are entered in the upper left hand corner of each cell.

d. Quantities of lubricating oil to ship from one storage location to a particular destroyer are entered within circles in the center of certain cells. The determination of these quantities is explained below; in the first tableau the circled quantities represent a first feasible solution;

e. Column (u) and row (v), which are left blank in the first tableau, will be utilized in later computations; these may be considered, simply, as two arbitrary variables. Results of computations with these variables will be entered in the lower right hand corner of later tableaux.

To begin iterations one must, as in the simplex method, find a first feasible solution (i.e., a solution which is not necessarily optimal but is a possible solution).

There are several ways available for obtaining a first feasible solution in transportation type problems; Hadley describes five different methods of getting the first feasible solution.[40]

One effective way of getting the first feasible solution is to scan the tableau and find the smallest shipping cost (if there are "ties" where two or more costs are equally the smallest, then anyone of the costs may be chosen arbitrarily). After the smallest cost has been determined one assigns as much as possible of the item being transported to that cell having the smallest cost. One then finds the next smallest cost and again assigns as much as possible of the item being transported to that cell having the second smallest cost. This procedure is followed (with the third smallest cost, etc.) until all destinations' demands are satisfied and all sources are exhausted. For example, in tableau (3.2a) the smallest cost is zero (0) in cell (3,2) and we assign as much as possible of the lubricating oil to this box (or 60 gal.). The next smallest cost is 1¢/gal. in cells (3,1), (2,2) and (1,5); we can assign 15 gal. (which is the quantity that can still be furnished from storage location 3 [75 gal. - 60 gal.]) to cell (3,1) and 25 gal. to cell (1,5), but we can assign nothing to cell (2,2) since the destination requirement (60 gal.) has already been filled

---

[40]Hadley. Op. Cit., p. 293 and pp. 304-309.

at cell (3,2). The next (or third smallest cost is 2¢/gal. in cells (1,2), (2,3), (2,5) and (3,5). Immediately we can rule out boxes (1,2), (2,5) and (3,5) since the destination requirements have already been filled. This leaves only cell (2,3) to which we can assign a quantity of 40 gal. We continue in this manner and finally obtain the first feasible solution as shown in tableau (3.2a).

We are now in a position to begin iterations toward a final optimal solution, in a step-by-step procedure.

<u>Step 1</u>. The vacant cells must be evaluated to determine if the feasible solution is optimal. To do this we use the two arbitrary variables (u) and (v), making the following definition:

$$c_{ij}^{B} = u_i + v_j$$

where: c_{ij}^{B} = the cost associated with the cells (i,j) in the feasible solution which have been assigned quantities.

Then, for any particular cell (i,j):

$$z_{ij} - c_{ij} = u_i + v_j - c_{ij}$$

Hadley gives a concise proof of this relationship.[41] By arbitrarily assigning one of the variables (u_i, v_j) some

---

[41] Hadley. <u>Op</u>. <u>Cit</u>., p. 310.

value we can solve for all $z_{ij} - c_{ij}$. For example, in tableau (3.2b) we have arbitrarily set $u_1 = 0$, then:

$$c_{11}^B = 3;\ u_1 = 0;\quad \text{then, } 3 = 0 + v_1;\ v_1 = 3$$

$$c_{21}^B = 4;\ v_1 = 3;\quad \text{then, } 4 = u_2 + 3;\ u_2 = 1$$

$$c_{31}^B = 1;\ v_1 = 3;\quad \text{then, } 1 = u_3 + 3;\ u_3 = -2$$

$$c_{32}^B = 0;\ u_3 = -2;\quad \text{then, } 0 = -2 + v_2;\ v_2 = 2$$

$$c_{23}^B = 2;\ u_2 = 1;\quad \text{then, } 2 = 1 + v_3;\ v_3 = 1$$

$$c_{24}^B = 4;\ u_2 = 1;\quad \text{then, } 4 = 1 + v_4;\ v_4 = 3$$

$$c_{15}^B = 1;\ u_1 = 0;\quad \text{then, } 1 = 0 + v_5;\ v_5 = 1$$

Once all u_i, and v_j have been determined, as shown above, we may evaluate the empty cells. For example:

$$z_{12} - c_{12} = u_1 + v_2 - c_{12} = 0 + 2 - 2 = 0$$

$$z_{33} - c_{33} = v_3 + v_3 - c_{33} = (-2) + 1 - 5 = -6$$

This procedure is followed until all empty cells are evaluated; the resulting evaluations are entered in the lower right hand corner of the empty cells in the tableaux.

<u>Step 2</u>. The tableau is next scanned to determine if any $z_{ij} - c_{ij}$ is positive. When any $z_{ij} - c_{ij} > 0$ an optimum solution has not been reached. If all $z_{ij} - c_{ij} \leq 0$ an optimum solution to the problem has been achieved. It

may be noted here that this criteria is just the opposite of that used in the simplex method. Some authors use a somewhat different procedure with the resulting criteria being if any evaluation of an empty cell is negative then an optimum has not been reached: (for example, see Garvin[42]). As long as one is consistent in the evaluations, the criteria is unimportant. The difference is pointed out here only as a caution that there are also variations of this procedure among the authorities, as there were differences in the simplex method.

Step 3. When an optimum solution has not been reached the empty cell containing the largest positive value of $z_{ij} - c_{ij}$ is chosen to be filled by some quantity presently filling some other cell (if "ties" occur among the largest value of $z_{ij} - c_{ij}$ any of the largest positive values may be chosen arbitrarily). Since the system must be in balance at all times, several adjustments must be made. The most efficient method of making these adjustments is as follows:

a. Place some identification (say: theta [θ]) in the cell to be filled (notice cell [2.2] in tableau [3.2b]).

b. If we add some amount θ to a row, then, to remain in balance the same amount θ must be subtracted from

---

[42]Garvin. Op. Cit., p. 95.

some other circled quantity in that row; but, from whatever column in the row we subtract θ, then, we must add θ to another row in that column; etc. In other words, we begin at the cell having the largest positive value of $z_{ij} - c_{ij}$ and add an amount θ to that cell, then, we progress from that cell alternatively subtracting and adding θ in cells having circled quantities until we return to the cell at which we began. For example, in tableau (3.2b) we place a (+θ) in cell (2,2); a (-θ) in cell (2,1); a (+θ) in cell (3,1); and a (-θ) in cell (3,2). Thus, we have completed the circuit. Notice that this route which has been followed is the only one possible. If we had put a (-θ) in cell (2,3) instead of cell (2,1) there would have been no other cell in column (3) to which we could have added a (+θ).

c. Once we have determined which cells have to be adjusted we find some cell having a negative adjustment (-θ) and which has the smallest circled quantity among all the negative (-θ) adjustment (in tableau [3.2b] it is cell [2,1]). Theta (θ) is then set equal to this smallest circled value and the respective additions and subtractions are made, with the results entered in corresponding cells of a new tableau. Cells which were not affected by theta (θ) adjustments are reproduced in the new tableau from the previous tableau in their respective positions.

One iteration has now been completed and to continue on to reach a final optimal solution we must return to step 1, and repeat the entire procedure again until all vacant cells have $z_{ij} - c_{ij} \leqq 0$. In this example problem we reach the optimal solution after two iterations (see tableaux [3.2c] and [3.2d]).

To interpret the final results we merely examine the final optimal tableau and read off the various quantities to be shipped from a source to a destination. In the example problem we see that the source S_2 (storage location 2, in row 2) ships 60 gallons of lubrication oil to destination D_2 (destroyer 2, in column 2). It is interesting to note that storage location S_3 gives nothing to destroyer D_2 even though the vessel must be tied up at S_3's pier because there is no transportation cost involved in shipping from S_3 to D_2.

To obtain the cost of a shipment one merely multiplies the cost (c_{ij}) associated with a cell and the quantity to be transported (circled quantity in the cell). The total cost of the optimal program is, obviously, the sum of all the individual shipping costs.

Row ↓ / Column →	1	2	3	4	5	S_i	u_i
1	3 75	2	3	4	1 25	100	
2	4 10	1	2 40	4 75	2	125	
3	1 15	0 60	5	3	2	75	
D_j	100	60	40	75	25		
v_j							

TOTAL COST

(75)(3¢)	=	$2.25
(25)(1¢)	=	0.25
(10)(4¢)	=	0.40
(40)(2¢)	=	0.80
(75)(4¢)	=	3.00
(15)(1¢)	=	0.15
(60)(0¢)	=	0.00
Total		$6.85

TABLEAU 3.2a

Row ↓ / Column →	1	2	3	4	5	S_i	u_i
1	3 75	2 0	3 −2	4 −1	1 25	100	0
2	4 $-\theta$ 10	1 $+\theta$ +2	2 40	4 75	2 0	125	1
3	1 $+\theta$ 15	0 $-\theta$ 60	5 −6	3 −2	2 −3	75	−2
D_j	100	60	40	75	25		
v_j	3	2	1	3	1		

TOTAL COST

(75)(3¢)	=	$2.25
(25)(1¢)	=	0.25
(10)(4¢)	=	0.40
(40)(2¢)	=	0.80
(75)(4¢)	=	3.00
(15)(1¢)	=	0.15
(60)(0¢)	=	0.00
Total		$6.85

TABLEAU 3.2b

Row \ Column	1	2	3	4	5	S_i	u_i
1	3 75 $-\theta$	2 0	3 0	4 $+\theta$ +1	1 25	100	0
2	4 -2	1 10 $+\theta$	2 40	4 75 $-\theta$	2 -2	125	-1
3	1 25 $+\theta$	0 50 $-\theta$	5 -4	3 0	2 -3	75	-2
D_j	100	60	40	75	25		
v_j	3	2	3	5	1		

<u>TOTAL</u> <u>COST</u>

(75)(3¢) = $2.25
(25)(1¢) = 0.25
(10)(1¢) = 0.10
(40)(2¢) = 0.80
(75)(4¢) = 3.00
(25)(1¢) = 0.25
(50)(0¢) = 0.00

Total $6.65

TABLEAU 3.2c

Row \ Column	1	2	3	4	5	S_i	u_i
1	3 25	2 0	3 0	4 50	1 25	100	0
2	4 -2	1 60	2 40	4 25	2 0	125	-1
3	1 75	0 0	5 -4	3 0	2 -1	75	-2
D_j	100	60	40	75	25		
v_j	3	2	3	5	3		

<u>TOTAL</u> <u>COST</u>

(25)(3¢) = $0.75
(50)(4¢) = 2.00
(25)(1¢) = 0.25
(60)(1¢) = 0.60
(40)(2¢) = 0.80
(25)(4¢) = 1.00
(75)(1¢) = 0.75

Total $6.15

Final (Optimal) Tableau

TABLEAU 3.2d

The "Stepping-Stone" (Transportation) Method

What is generally called the transportation (or distribution) method is associated with the "stepping-stone" technique (which is another method of evaluating the vacant cells in the transportation tableaux). This one difference distinguishes the transportation method from the MODI method. Most authors of linear programming texts give comprehensive treatments of the "stepping-stone" method (for example, see Charnes and Cooper[43]).

The transportation method, using the "stepping-stone" technique and the MODI method, using the evaluation technique described above, obtain the same numerical results. It is believed, however, that the "stepping-stone" method is more difficult to work with and is more apt to cause a person to make more calculating errors.

Degeneracy

Degeneracy may be encountered in either the simplex algorithm or in the various distribution algorithms.

In the simplex method, degeneracy is caused by more than one vector passing through a single point, and it is manifested when, in trying to determine which variable to eliminate from the basis, "ties" occur among the smallest

---

[43] Charnes and Cooper. Op. Cit., pp. 41-49.

ratios. In the case of "ties", regardless of which variable is selected to leave the basis, the other variables involved in the tie will be reduced to zero, and there will be no change in the functional (Z). In general, the procedure for breaking ties, as given above, will be sufficient to allow the procedure to continue to a final optimal solution. It is possible, however, that the computations will cycle, or, in other words, after several iterations, some previously eliminated variable re-enters the basis. It should be obvious, then, that cycling and degeneracy are not synonymous. Degeneracy is necessary before cycling can occur, but degeneracy is not sufficient to cause it. Garvin states that ". . .while degeneracy is common, cycling is extremely rare."[44] Although there have been problems devised to cycle, Gass reports ". . .there have not been any practical problems that have been known to cycle."[45] Since degeneracy and cycling in the simplex method are of no practical concern the method of solution for the simplex algorithm described above would appear sufficient; however, there are procedures designed to absolutely prevent

---

[44]Garvin. Op. Cit., p. 47.

[45]Gass, Saul I., Linear Programming Methods and Applications, New York: McGraw-Hill Book Co., Inc., 1958. P. 103.

cycling and these may be found in the literature (for example, see: Gass,[46], Garvin,[47], or Hadley[48]).

Degeneracy may occur in the various "transportation" methods whenever there are less than m + n - 1 cells of the tableau filled with circled quantities (m = number of sources or rows; n = number of destinations or columns). This situation may occur at any stage of the computation and may be overcome by a very simple procedure: one simply fills sufficient vacant cells with zeros in positions which will prevent degeneracy. The cells containing zeros are then treated as quantities to be shipped (circled quantities). Hadley presents a similar method which prevents degeneracy in transportation computations, from the outset.[49]

It is theoretically possible for a transportation type computation to cycle; however, it would appear that cycling is even less of a problem, here, than in the simplex algorithm. Hadley states: ". . .no transportation problem has ever been known to cycle."[50]

---

[46] Ibid., chapter 7.

[47] Garvin. Op. Cit., chapter 14.

[48] Hadley, Op. Cit., chapter 6.

[49] Ibid., p. 302.

[50] Ibid., p. 299.

CHAPTER IV

GENERAL APPLICATIONS OF LINEAR PROGRAMMING

General

Examples of linear programming applied to various problems of industrial management are legion. Perhaps one of the reasons for such wide-spread application of linear programming in industrial enterprises is the relative ease of data quantification such as costs, profits, mix ratios, etc. In addition it is a generally recognized fact that for management decision making, complete data accuracy is not always imperative; estimation of data is, in many cases, acceptable and adds, therefore, to the relative ease of the data quantification.

In this chapter a survey of the generalized classical applications of linear programming will be presented together with five specific numerical examples. It is intended only that this should indicate the manner in which linear programming may be applied, and it is not intended to be a complete representation of all the applications to which linear programming may be put.

Classical Applications of Linear Programming

There has been a tendency in the literature to classify applications of linear programming by means of

sterotypes. In one sense there is nothing wrong with such a procedure; in fact it aids classification of applications considerably. On the other hand, however, when an idea, concept or application (as being discussed, here) is stereotyped, it is generally very difficult to remove the idea, concept, or application from one viewpoint and put in into a different perspective and setting. The real worth of ideas, concepts, and applications, generally, lies in their versatility.

In this section some of the sterotyped classifications of linear programming applications will be pointed out, together with some possible diverse uses of the particular applications.

(Diet Problem)

The first classical linear programming problem deals with the feeding of a group of individuals (say, Navy personnel aboard ship) where it is desired that the personnel be given meals meeting all nutritional requirements but with food of minimum cost. Thus, if we let:

X_i = i th food item in the diet

c_i = cost of food item X_i

a_{ij} = nutritional content (of nutrient b_j) in food item X_i

b_j = j th nutrient required in a diet.

Then:

$$\text{Objective Function:} \quad \sum_{i=1}^{n} c_i X_i = Z \text{ (Min)}$$

$$\text{Constraints:} \quad \sum_{i=1}^{n} a_{ij} X_i \geqq b_j \quad (j=1,2,3,\ldots,m)$$

$$X_i \geqq 0 \quad (i=1,2,3,\ldots,n)$$

Although a linear programming approach to meal planning will indicate how to feed a group of Navy personnel with food containing the required nutritional elements at minimum cost to the government, it is not a generally recommended method of general mess management since it may lead to some rather unpalatable meals. Linear programming, being very objective and single-minded, can indicate nothing about food variety, nor can personal food preferences (among Navy personnel) be optimized.

The diet problem was the forerunner of all the blending or mixing type problems of which the widest application has been in the petroleum refining industry relative to the blending of gasoline. A numerical example of a gasoline blending problem is given in the following section of this chapter.

The diet problem may also be applied in situations such as the manufacturing of any product (from ice cream or sausage to paint or steel) where a minimum cost of production is desired and there are certain minimum raw

material requirements; or, restrictions on the availability of raw materials; or, restrictions on the quality of the product; or, restrictions on the quantities of output of the product which may be produced; etc. Generally these problems take the form of having an infinite number of ways in which the raw materials can be combined to make the output product.

Blending, or mixing, type problems are not limited to inanimate product manufacturing. Another aspect of blending was described in chapter II when the output of a disbursing clerk was maximized and his time was "blended" or "mixed" in working on pay records and vouchers. Similarly the utilization of equipment could be handled in this way.

(Transportation Problem)

The transportation problem was the second type of linear programming problem to be specified and, perhaps, is one of the most, if not the most important linear programming problem type. Its importance lies in the fact that a relatively simplified algorithm is available (the MODI method) for solution of transportation type problems and has a great range of applications. Since the generalized form of the transportation problem was given in chapter III it will not be repeated here.

One application of the transportation problem was indicated in chapter III; that of simply finding the least cost of transporting a homogeneous product from several storage locations to several destinations where the total demands for the item, at the destinations, exactly equalled the total availability of the item at all the storage locations. There are several variations of this one application:

a. Availability of the item exceed the demand for it;

b. Demands for the item exceed the availability of it;

c. Only partial quantities can be supplied from one storage location to some destination. (A numerical example of this variation is given in the following section of this chapter.)

d. Items can, or must, be transhipped between some destinations.

Another application of the transportation problem, of importance in the Navy, is the routing of sea going vessels (say fleet tankers). In this application the tankers are located at various ports, anywhere throughout the world, and they are required at some other ports to transport fuel to some other locations. The problem, obviously, is to find the optimum assignment of tankers to load the fuel and transport it at minimum cost to the government.

One other application of the transportation problem is its use in bid evaluation. Here, several bidders propose to supply the government (at various government storage locations) some required item, at various costs. By utilizing the transportation type problem approach, assignment of contracts to the various bidders may be made at least cost to the government.

(Caterer Problem)

The classical presentation of the caterer problem concerns the manager of a restaurant (for example), or a caterer, who knows that on specific days in the future, while serving meals, he will have various requirements for napkins. To begin his business he will have to buy an initial quantity of napkins but thereafter, the manager, or caterer, may either buy more napkins or send the soiled ones to laundries which can return clean napkins in specified times at specified costs (with faster service naturally costing more money). The problem, then, is to find the optimum program which will give an adequate supply of napkins at all times at the least cost.

The generalization of this problem may be accomplished as follows:

Let:

R_j = clean napkins required on the j th day ($j=1,2,3. .,n$)

p = days required for slow laundry service

q = days required for fast laundry service

$q < p$

a = cost of new napkins

b = cost of laundrying one napkin with slow service

c = cost of laundrying one napkin with fast service

$a > c > b$

x_j = napkins purchased on the j th day

y_j = napkins sent to the laundry with slowest service, on the j th day

z_j = napkins sent to the laundry with fastest service, on the j th day

s_j = soiled napkins stored on the j th day and not sent to any laundry.

Then:

$$x_j + y_{j-p} + z_{j-q} = R_j$$

and

$$y_j + z_j + s_j - s_{j-1} = R_j$$

The objective function is then:

$$\sum_{j=1}^{n} (ax_j + by_j + cz_j) = Z \text{ (Min)}$$

A numerical example (presented as the Wardroom Steward problem) is given in the following section.

The caterer problem may be applied to procurement and maintenance problems. For example, if a supply officer, in order to meet future specific demands, has to determine whether a certain type of spare part should be either purchased or repairable items sent to various repair shops (assuming other repair criteria has been met) then we have an analogy between napkins and spare parts.

The storage parameter (s_j) may be assigned some cost, also, and then the caterer problem may be used to determine the program to optimize over-production (or, excess procurement) in one period (with attendant storage requirements) and future demands for some item. In this manner production (or procurements) may be "smoothed" when there are seasonal fluctuations in requirements for the item.

Comment on the Computer Solution of Linear Programming Problems

The numerical examples contained in the following sections were solved by means of an IBM 1620 computer, at the University of Kansas. The program used in obtaining solutions was from the 1620 General Program Library, titled Linear Programming II (10.1.008), by F. W. Wood. The program is a five (5) phase program written in "Fortran with Format" language.

Although there are published instructions for the program which was used (10.1.008), these instructions are dependent upon the published instructions for the program titled Linear Programming I (10.1.007); both sets of instructions are necessary to successfully work with Linear Programming II (10.1.008).

The program negates the functional and, therefore, actually solves a minimizing problem. To solve a maximizing problem the objective function first has to be negated, prior to solving.

Since the library program instructions are not specific regarding the times required for solution of problems it may be of interest to note the times which were required for solution of the numerical examples which follow:

Problem	No. of Variables	No. of Equations	Iterations Required	Approximate Time Required (Min.)
Gasoline Blending	22	10	14	10
Distribution (Balanced supply & demand)	40	14	36	15
Distribution (Unbalanced supply & demand)	44	15	40	23
Wardroom Steward	40	20	25	15
Wardroom Steward (Modification no. 1)	50	20	26	18

Numerical Examples of Linear Programming Problems

This section contains five numerical examples from the general classifications of linear programming problems presented in a previous section of this chapter. Although the numerical examples are somewhat more involved than have been presented thus far, they are considerably simpler than they would otherwise be in real life situations. Nevertheless, the following examples are intended to indicate: (a) how a problem may be put into the farmework of a mathematical model; (b) how the mathematical model may be put into a proper format for computer solution; and (c) finally, how the computer solution may be interpreted.

(Gasoline Blending Problem)[51]

A company wishes to make as much income as it can from its refinery operations. It is believed that the manufacture of aviation gasoline will be profitable. It is proposed that three types of aviation gasoline (types A, B, and C) should be made. In order to do this, raw materials consisting of: alkylate, catalytic cracked gasoline, straight-run gasoline, and isopentane, are to be blended to make the aviation gasolines A, B, and C; any

---

[51]Data adapted from Gale. Op. Cit., pp. 64-65.

excess of the materials will go into gasoline for automobile use. The properties of the above raw material components are given below in table (4.1).

TABLE 4.1

GASOLINE BLENDING PROBLEM
RAW MATERIAL COMPONENTS

Component	Reid Vapor Pressure	Octane No. (with 0.5 ml/gal. Tetraethyl-lead content)	Octane No. (with 4 ml/gal. Tetraethyl-lead)	Availability (bbls/day)
Alykylate	5	94	107.5	3,800
Catalytic cracked gasoline	8	83	93.0	2,652
Straight run gasoline	4	74	87	4,081
Isopentane	20.5	95	108	1,300

The raw material requirements and expected revenue of the various products are given below:

TABLE 4.2

GASOLINE BLENDING PROBLEM
RAW MATERIAL REQUIREMENTS

Product	Reid Vapor Pressure	Tetraethyl-lead level	Octane No.	Income ($/bbl)
Av. Gas. A	$\leqq$ 7	0.5	$\geqq$ 80	4.908
Av. Gas. B	$\leqq$ 7	4.0	$\geqq$ 91	5.437
Av. Gas. C	$\leqq$ 7	4.0	$\geqq$ 100	6.042
Auto Gas.	---	---	---	4.548

The blending process is shown in figure (4.1).

The following constraining relationships may be formed by utilizing the data found in tables (4.1) and (4.2):

Reid vapor pressure constraint:

AV. GAS. "A"

$$5X_1 + 8X_2 + 4X_3 + 20.5X_4 \leqq 7(X_1 + X_2 + X_3 + X_4)$$

or $$-2X_1 + X_2 - 3X_3 + 13.5X_4 \leqq 0$$

AV. GAS. "B"

$$5X_5 + 8X_6 + 4X_7 + 20.5X_8 \leqq 7(X_5 + X_6 + X_7 + X_8)$$

or $$-2X_5 + X_6 - 3X_7 + 13.5X_8 \leqq 0$$

AV. GAS. "C"

$$5X_9 + 8X_{10} + 4X_{11} + 20.5X_{12} \leqq 7(X_9 + X_{10} + X_{11} + X_{12})$$

or $$-2X_9 + X_{10} - 3X_{11} + 13.5X_{12} \leqq 0$$

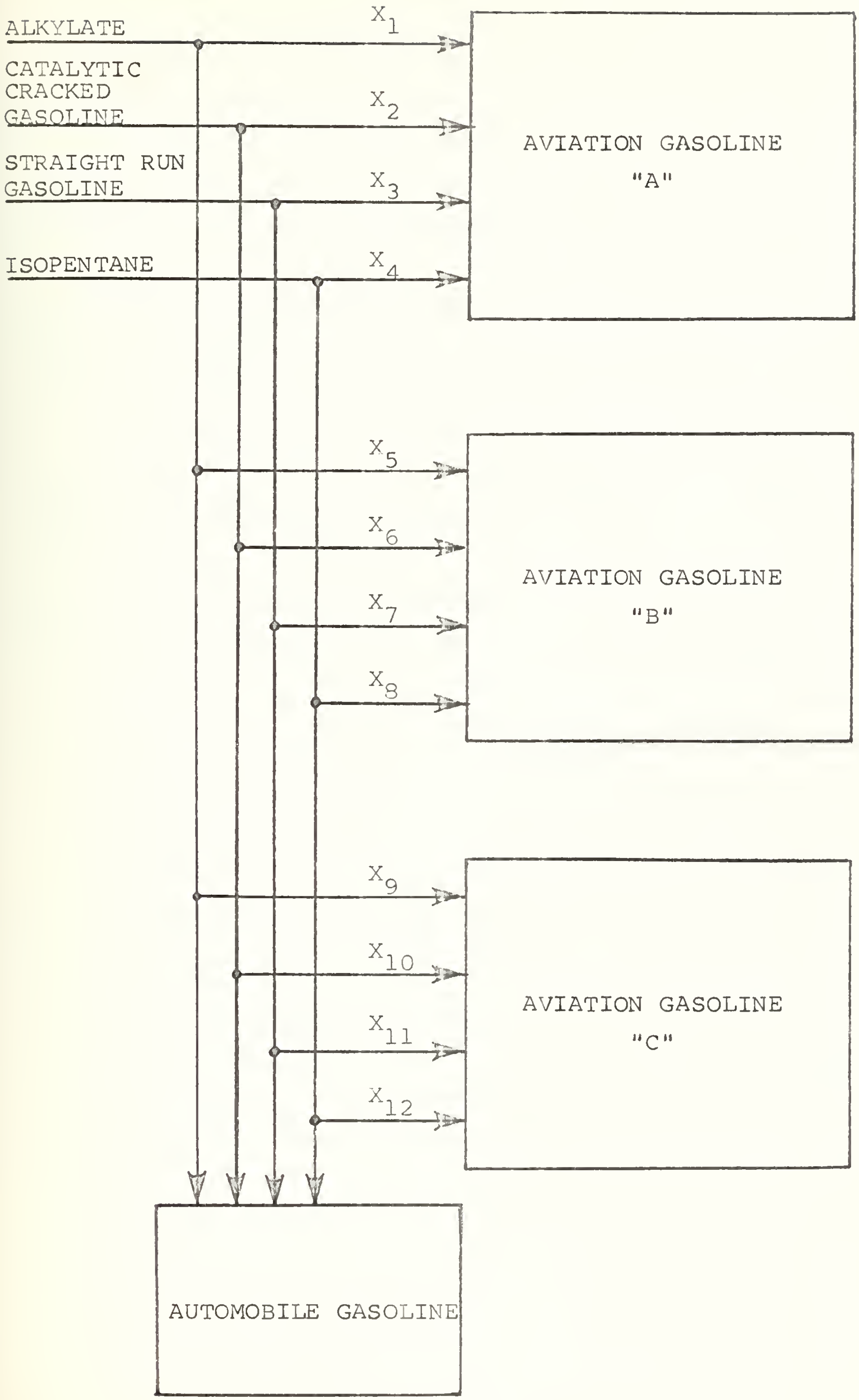


FIGURE 4.1

SCHEMATIC DIAGRAM OF THE GASOLINE BLENDING PROBLEM

Octane Number constraints:

AV. GAS. "A"

$$94X_1 + 83X_2 + 74X_3 + 95X_4 \geqq 80(X_1+X_2+X_3+X_4)$$

or $$14X_1 + 3X_2 - 6X_3 + 15X_4 \geqq 0$$

AV. GAS. "B"

$$107.5X_5+93X_6+87X_7 + 108X_8 \geqq 91(X_5+X_6+X_7+X_8)$$

or $$16.5X_5+ 2X_6- 4X_7 + 17X_8 \geqq 0$$

AV. GAS. "C"

$$107.5X_9+93X_{10}+87X_{11}+108X_{12} \geqq 100(X_9+X_{10}+X_{11}+X_{12})$$

or $$7.5X_9- 7X_{10}-13X_{11}+ 8X_{12} \geqq 0$$

Raw material availability constraints:

$$X_1 + X_5 + X_9 \leqq 3,800$$

$$X_2 + X_6 + X_{10} \leqq 2,652$$

$$X_3 + X_7 + X_{11} \leqq 4,081$$

$$X_4 + X_8 + X_{12} \leqq 1,300$$

The total volume of liquid is 11,833 bbl/day (3,800 + 3,652 + 4,081 + 1,300). If we let the respective volumes of aviation gasoline A, B, and C which is produced be:

V_A, V_B, V_C.

Then:

$$V_A = X_1 + X_2 + X_3 + X_4$$

$$V_B = X_5 + X_6 + X_7 + X_8$$

$$V_C = X_9 + X_{10}+ X_{11}+ X_{12}$$

The automobile gasoline produced (V_{AG}) is, then:

$$V_{AG} = 11{,}833 - (V_A + V_B + V_C)$$

The objective function to be maximized, then, is:

$$4.908V_A + 5.437V_B + 6.042V_C + 4.548(11{,}833 - V_A - V_B - V_C) = Z \text{ (Max.)}$$

or:

$$53{,}816.484 + 0.360V_A + 0.889V_B + 1.494V_C = Z \text{ (Max.)}$$

Temporarily the constant (53,816.484) may be ignored and we have the objective function and mathematical model as follows (after insertion of appropriate slack variables):

OBJECTIVE FUNCTION:

$$0.360(X_1 + X_2 + X_3 + X_4) + 0.889(X_5 + X_6 + X_7 + X_8)$$
$$+ 1.494(X_9 + X_{10} + X_{11} + X_{12}) = Z' \text{ (Max.)}$$

Note that the objective function shown above is negated when preparing the input data for the computer program.

CONSTRAINING EQUATIONS:

$$- 2X_1 + X_2 - 3X_3 + 13.5X_4 + X_{13} = 0$$

$$14X_1 + 3X_2 - 6X_3 + 15.0X_4 - X_{14} = 0$$

$$- 2X_5 + X_6 - 3X_7 + 13.5X_8 + X_{15} = 0$$

$$16.5X_5 + 2X_6 - 4X_7 + 17.0X_8 - X_{16} = 0$$

$$- 2X_9 + X_{10} - 3X_{11} + 13.5X_{12} + X_{17} = 0$$

$$7.5X_9 - 7X_{10} - 13X_{11} + 8.0X_{12} - X_{18} = 0$$

$$X_1 + X_5 + X_9 + X_{19} = 3{,}800$$

$$X_2 + X_6 + X_{10} + X_{20} = 2{,}652$$

$$X_3 + X_7 + X_{11} + X_{21} = 4{,}081$$

$$X_4 + X_8 + X_{12} + X_{22} = 1{,}300$$

The constant which has been ignored in the objective function represents the income which would be obtained if all $x_j = 0$; or, in other words if no aviation gasoline were made at all. After solution of the above system it must be remembered that the constant has to be added to the income indicated for the functional since the functional will indicate only the additional income derived solely from the manufacture of the aviation gasolines.

Diagrammatically the solution is shown below in figure (4.2).

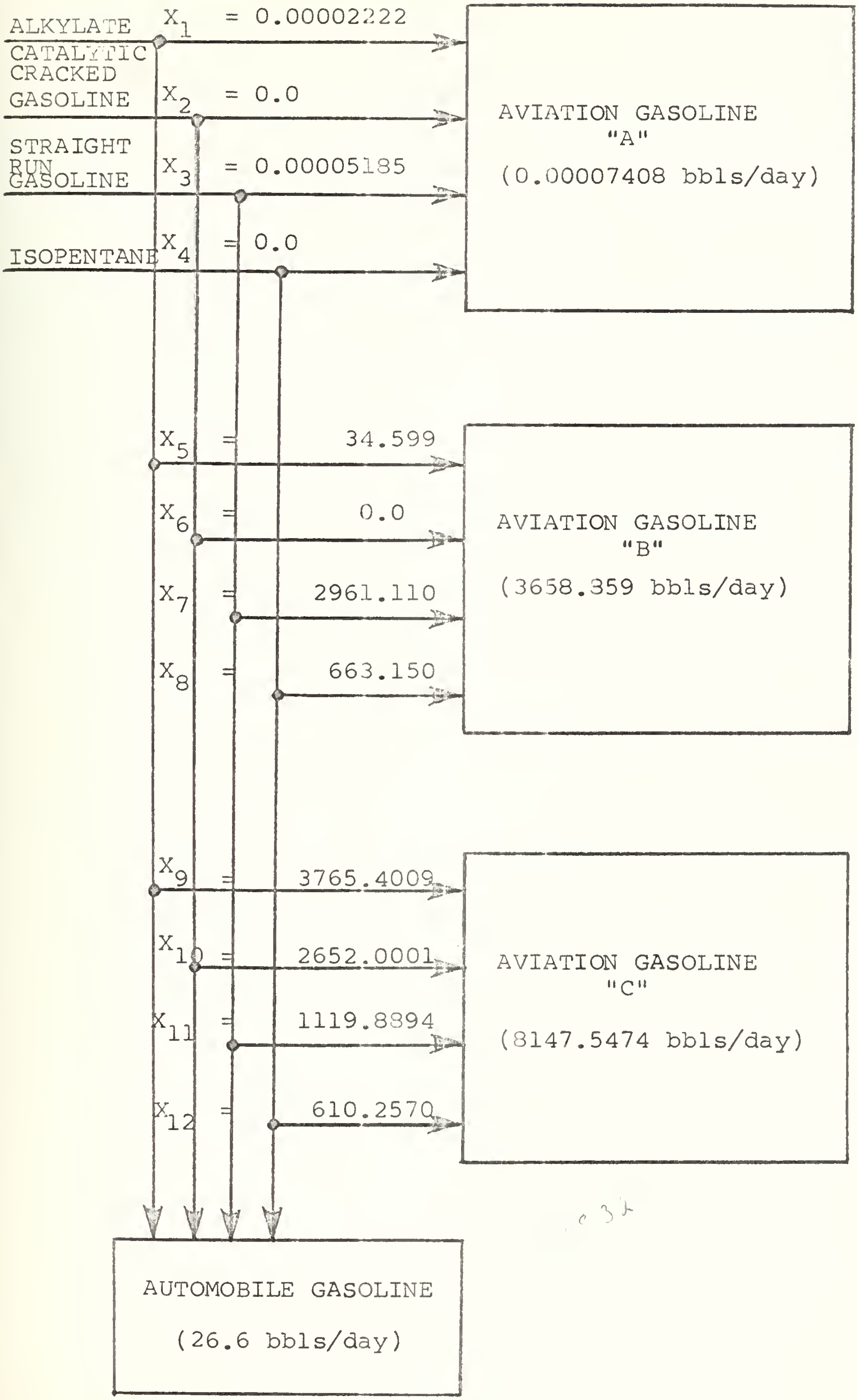


FIGURE 4.2

SOLUTION OF THE GASOLINE BLENDING PROBLEM

The functional, as given in the computer output, is \$15,425.161; adding the constant term \$53,816.484, the total income, then, is \$69,241.645. It can be noted, however, that the solution calls for a minute quantity of aviation gasoline "A", to be produced, which would be neglected in reality, as being an unprofitable product. By neglecting all production of aviation gasoline "A" our functional remains the same, however, since the small quantities called for in the optimal solution add only \$0.000024 and cannot be detected in the final solution.

(Distribution Problem)[52]
(Balanced Supply and Demand)

A U. S. Navy Supply Officer has an item (for example, say cement, but the product is really immaterial) which is stored at four depots. He receives requisitions (or demands) from ten requisitioners (or customers) for various quantities of the cement and the officer knows what shipping costs are involved when shipping cement from any of the four depots to any of the ten requsitioners. These data are shown in table (4.3). Each cell of table (4.3) must be described by a variable because each cell is associated with a specific shipping cost and represents some potential

---

[52]Data adapted from: Hetrick, James C., "Mathematical Models in Capital Budgeting," Harvard Business Review, (January-February, 1961) reprinted in HBR Statistical Decision Series, part II, p. 67.

quantity of cement to be shipped from a depot to a requisitioner. In the final solution the variables will indicate which depot should supply which requisitioners and in what quantity. Then, variables may be assigned as shown in table (4.4).

Recalling the generalized transportation problem formulation shown in chapter III, a mathematical model is constructed as follows:

TABLE 4.3

SHIPPING COSTS OF THE DISTRIBUTION PROBLEM
(BALANCED SUPPLY AND DEMAND)

Shipping Costs ($/cwt)										Qty. Avail. (cwt)	Storage Location (Depot)
1.45	1.15	0.21	2.54	1.98	0.49	1.00	0.36	2.35	3.00	6500	1
1.67	1.45	0.50	2.80	2.15	0.80	1.26	0.75	2.75	3.25	2000	2
1.48	1.20	0.30	2.59	2.00	0.60	1.10	0.55	2.40	3.00	4800	3
0.85	0.65	0.10	2.00	1.25	0.38	0.89	0.25	1.75	2.25	4800	4
2000	2400	1500	1300	1300	2100	1800	1700	2200	1800	Quantity Requisitioned (cwt)	
1	2	3	4	5	6	7	8	9	10	Requisitioner	

TABLE 4.4

VARIABLES OF THE DISTRIBUTION PROBLEM
(BALANCED SUPPLY AND DEMAND)

VARIABLES										Qty. Avail. (cwt)	Storage Location (Depot)
X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	6500	1
X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	X_{16}	X_{17}	X_{18}	X_{19}	X_{20}	2000	2
X_{21}	X_{22}	X_{23}	X_{24}	X_{25}	X_{26}	X_{27}	X_{28}	X_{29}	X_{30}	4800	3
X_{31}	X_{32}	X_{33}	X_{34}	X_{35}	X_{36}	X_{37}	X_{38}	X_{39}	X_{40}	4800	4
2000	2400	1500	1300	1300	2100	1800	1700	2200	1800	Quantity Requisitioned	
1	2	3	4	5	6	7	8	9	10	Requisitioner	

OBJECTIVE FUNCTION:

$$1.45X_1 + 1.15X_2 + 0.21X_3 + 3.54X_4 + 1.98X_5 + 0.49X_6 + 1.00X_7 +$$
$$0.36X_8 + 2.35X_9 + 3.00X_{10} + 1.67X_{11} + 1.45X_{12} + 0.50X_{13} + 2.80X_{14} +$$
$$2.15X_{15} + 0.80X_{16} + 1.26X_{17} + 0.75X_{18} + 2.75X_{19} + 3.25X_{20} + 1.48X_{21} +$$
$$1.20X_{22} + 0.30X_{23} + 2.59X_{24} + 2.00X_{25} + 0.60X_{26} + 1.10X_{27} + 0.55X_{28} +$$
$$2.40X_{29} + 3.00X_{30} + 0.85X_{31} + 0.65X_{32} + 0.10X_{33} + 2.00X_{34} + 1.25X_{35} +$$
$$0.38X_{36} + 0.89X_{37} + 0.25X_{38} + 1.75X_{39} + 2.25X_{40} = Z \text{ (Min.)}$$

Note that this objective function does not have to be negated prior to preparing the computer input data; since the computer program is written to negate the objective function automatically it, then, causes the computer to solve for the minimum.

Constraints:

Supply (or row) restrictions:

$$\sum_{j=1}^{10} X_j = 6,500$$

$$\sum_{j=11}^{20} X_j = 2,000$$

$$\sum_{j=21}^{30} X_j = 4,800$$

$$\sum_{j=31}^{40} X_j = 4,800$$

Demand (or column) restrictions:

$$X_1 + X_{11} + X_{21} + X_{31} = 2{,}000$$

$$X_2 + X_{12} + X_{22} + X_{32} = 2{,}400$$

$$X_3 + X_{13} + X_{23} + X_{33} = 1{,}500$$

$$X_4 + X_{14} + X_{24} + X_{34} = 1{,}300$$

$$X_5 + X_{15} + X_{25} + X_{35} = 1{,}300$$

$$X_6 + X_{16} + X_{26} + X_{36} = 2{,}100$$

$$X_7 + X_{17} + X_{27} + X_{37} = 1{,}800$$

$$X_8 + X_{18} + X_{28} + X_{38} = 1{,}700$$

$$X_9 + X_{19} + X_{29} + X_{39} = 2{,}200$$

$$X_{10} + X_{20} + X_{30} + X_{40} = 1{,}800$$

The above system is shown in the canonical form of linear programming statement and is suitable for submission to the simplex algorithm rather than the transportation algorithm.

The solution for this problem is shown in table (4.5), below:

TABLE 4.5

SOLUTION OF THE DISTRIBUTION PROBLEM
(BALANCED SUPPLY AND DEMAND)

											Depot
200		500			2100		1700	2200		6500	1
100						1800				2000	2
1700	2400	1000	1300						1800	4800	3
2000	2400	1500	1300	1300	2100	1800	1700	2200	1800		
1	2	3	4	5	6	7	8	9	10	Requisitioner	

TOTAL COST: $23,113.00

It is interesting to note that not all of the requisitioners are supplied by the least expensive method of transportation; for example, it seems that it would be less expensive to supply requisitioner No. 3 from depot No. 4, or customer No. 6 from depot No. 4. It is to be emphasized that the solution, as given above, optimizes the system, not individual shipments. To optimize shipments to individual requisitioners, or shipments from individual depots would lead to sub-optimization of the whole system and, therefore, require greater overall cost.

(Distribution Problem)[53]
(Unbalanced Supply and Demand)

Suppose, now, that in the previous distribution problem the carrier who transports the cement from depot No. 1 to requisitioner No. 6 can handle only 1,000 cwt. because some of his equipment has broken down. (The previous solution calls for a delivery of 2,100 cwt. from depot No. 1 to requisitioner No. 6.) Since the requisitioner needs his material immediately, the question is how can the demand be filled, right now?

To solve this problem we can proceed as before but this time we must limit the quantity going from depot No. 1 to requisitioner No. 6 to 1,000 cwt. To do this we change the demand in column 6 from 2,100 cwt. and add one additional column having a demand of 1,100 cwt. The new column will have, in rows 1, 2, 3, and 4, the new variables X_{41}, X_{42}, X_{43}, and X_{44}, respectively. The new variable X_{41} (a partial shipment from depot No. 1 to requisitioner No. 6) represents the quantity of cement which the carrier cannot handle; therefore, in order to make sure variable X_{41} does not enter into the final solution it must be given a prohibitively high cost (say $10.00 per cwt.). Variables

[53]*Ibid.*

X_{42}, X_{43}, and X_{44} are identical, respectively, to variables X_{16}, X_{26}, and X_{36}, therefore, they must have the same transportation costs.

Now the problem is the same as before. To formulate the model we simply add the new variables as part of the previous objective function:

$$Z + 10.00X_{41} + 0.80X_{42} + 0.60X_{43} + 0.38X_{44} = Z' \text{ (Min)}$$

Then, to the supply (or row) restrictions we must add the new variables, as appropriate:

$$\sum_{j=1}^{10} X_j + X_{41} = 6,500$$

$$\sum_{j=11}^{20} X_j + X_{42} = 2,000$$

$$\sum_{j=21}^{30} X_j + X_{43} = 4,800$$

$$\sum_{j=31}^{40} X_j + X_{44} = 4,800$$

One additional demand (or column) restriction must be added and the restriction on column 6 must be changed from 2,100 to 1,000, while all the other column restrictions remain unchanged as before:

.

.

.

$$X_6 + X_{16} + X_{26} + X_{36} = 1{,}000$$

.

.

.

$$X_{41} + X_{42} + X_{43} + X_{44} = 1{,}100$$

The solution of this new problem is shown below in table (4.6):

TABLE 4.6

SOLUTION OF THE DISTRIBUTION PROBLEM
(UNBALANCED SUPPLY AND DEMAND)

												Depot
		1500			1000	100	1700	2200			6500	1
300						1700					2000	2
	2400		1300							1100	4800	3
1700				1300					1800		4800	4
2000	2400	1500	1300	1300	1000	1800	1700	2200	1800	1100		
1	2	3	4	5	6	7	8	9	10	11	Requisitioner	

TOTAL COST = $23,137.00

It can be seen that this new problem has increased transportation costs by \$24.00 (\$23,137 - \$23,113) but under the circumstances it is the optimum solution.

It is interesting to note, in this problem, as before, not all of the requisitioners are receiving supplies by the least individual cost, and that by changing the route of supply to requisitioner No. 6, some of the other supply routes also changed.

As mentioned previously this problem can be extended to cover other situations which occur in distribution systems. For example, it is not imperative that the supply source availability exactly equal the demands from the requisitioners.

If availability of the resource at the depots exceed requirements by the requisitioners then an extra column is added, as above. The extra column represents a fictitious (or dummy) requisitioner who is requisitioning all of the excess from the depots. The total for the dummy column, then, will be the difference between the total quantity of all depots and the total quantity being requisitioned by all the real (or actual) requisitioners. Since the dummy requisitioner does not exist and the excess material will not be moved from any depot the shipping costs assigned to the variables in the dummy column must all be zero.

Similiarly, if demands from requisitioners exceed availability of the resource at all the various depot, then an additional row (dummy depot) may be added with an "availability" equal to the total deficiency of all the actual depots. Shipping costs associated with the dummy depot must be exorbitently high to prevent any "bogus" shipments from the dummy depot entering the solution until all the least expensive modes of transportation have exhausted the actual available supply. "Bogus" shipments will appear in the solution and will represent quantities of the resource which cannot be shipped to meet certain demands; for this reason the functional obtained in the computer solution will not be valid and an actual shipping cost must be computed by hand.

(Wardroom Steward Problem)[54]

The Chief Steward aboard a large Navy vessel just being commissioned is faced with an optimizing problem. The wardroom has not yet been used and no napkins are on board. The Chief knows that the shipboard laundry will not be in operation for ten more days and that during this time meals will be served in the wardroom and he knows his napkin requirements will be as follows:

---

[54]Data adapted from Garvin. Op. Cit., pp. 119-120.

DAY	1	2	3	4	5	6	7	8	9	10
NAPKIN REQUIREMENT	50	60	80	70	50	60	90	80	50	100

The Chief Steward knows he can purchase napkins for 10¢ each ($a = .10$), and one laundry will launder soiled napkins for 1¢ each ($b = .01$) but it takes three days ($p = 3$) to get them back; another laundry will launder the soiled napkins for 3¢ each ($c = .03$) and return them in two days ($q = 2$). The chief steward must determine what program to follow in meeting the napkin requirement at a minimum cost to the wardroom officers' mess.

Recalling the problem generalized formulation presented in an earlier section of this chapter, we can construct the mathematical model as follows:

Napkin usage equations:

Day	Napkins Purchased		Napkins Returned (Fast)		Napkins Returned (Slow)		Napkins Required
	(x)	+	(z)	+	(y)	=	(R)
1	X_1					=	50
2	X_2					=	60
3	X_3	+	X_{11}			=	80
4	X_4	+	X_{12}			=	70
5	X_5	+	X_{13}	+	X_{22}	=	50
.	.		.		.		.
.	.		.		.		.
.	.		.		.		.
.	.		.		.		.
.	.		.		.		.
.	.		.		.		.
10	X_{10}	+	X_{18}	+	X_{27}	=	100

Napkin laundry equations:

Day	Napkins sent to Fast Laundry (z)		Napkins sent to Slow Laundry (y)		Napkins Stored $(s_j - s_{j-1})$		Napkins Used
1	X_{11}	+	X_{21}	+	$X_{31} - 0$	=	50
2	X_{12}	+	X_{22}	+	$X_{32} - X_{31}$	=	60
3	X_{13}	+	X_{23}	+	$X_{33} - X_{32}$	=	80
4	X_{14}	+	X_{24}	+	$X_{34} - X_{33}$	=	70
5	X_{15}	+	X_{25}	+	$X_{35} - X_{34}$	=	50
.	.		.		. .		.
.	.		.		. .		.
.	.		.		. .		.
.	.		.		. .		.
.	.		.		. .		.
.	.		.		. .		.
10	X_{20}	+	X_{30}	+	$X_{40} - X_{39}$	=	100

The objective function, then, is:

$$0.10(X_1 + \ldots + X_{10}) + 0.03(X_{11} + \ldots + X_{20}) + 0.01(X_{21} + \ldots + X_{30}) = Z \text{ (Min)}$$

The solution of the problem is shown below in table (4.7).

TABLE 4.7

SOLUTION OF THE WARDROOM STEWARD PROBLEM

DAY	NAPKINS PURCHASED	NAPKINS RETURNED FROM FAST LAUNDRY	NAPKINS RETURNED FROM SLOW LAUNDRY	NAPKINS USED DURING THE DAY	NAPKINS SENT TO FAST LAUNDRY	NAPKINS SENT TO SLOW LAUNDRY	NAPKINS STORED
1	50			50		50	
2	60			60	10	50	
3	80			80		60	20
4	10	10	50	70		90	
5			50	50		50	
6			60	60	30	30	
7			90	90	20	70	
8		30	50	80	30		50
9		20	30	50			50
10		30	70	100			100

TOTAL COST = $26.70

It can be noted that, on the third day, twenty soiled napkins are stored, overnight, and not sent to any laundry until the following day. The solution would have been the same had the twenty napkins been sent to the slow laundry on the third day and had then been stored as clean napkins on the sixth day when they were returned. In any case twenty napkins have to be stored in one form or another. Also, of course, the storage of napkins on the eighth, ninth and tenth days is in preparation for termination of the problem.

Although this problem spans only a period of ten days it could be extended for whatever period of time one desires (assuming, of course, one has computational capacity to handle extended problems).

(Wardroom Steward Problem)
(Modification No. 1)

Assume, now, that the Chief Steward finds there is another laundry which, for 4¢ each, will launder napkins and return them in one day. The Chief Steward wonders if he can improve his previous program by taking advantage of this new laundry's service.

To solve this new problem we need only to add an additional range of variables (say: w) describing the new service; this may be done as follows:

Napkin usage equations:

Day	Napkins Purchased		Napkins Returned (Fast)		Napkins Returned (Slow)		Napkins Returned (Fastest)		Napkins Required
	(x)	+	(z)	+	(y)	+	(w)	=	(R)
1	X_1							=	50
2	X_2					+	X_{41}	=	60
3	X_3	+	X_{11}			+	X_{42}	=	80
4	X_4	+	X_{12}	+	X_{21}	+	X_{43}	=	70
5	X_5	+	X_{13}	+	X_{22}	+	X_{44}	=	50
.	.		.		.		.		.
.	.		.		.		.		.
.	.		.		.		.		.
.	.		.		.		.		.
10	X_{10}	+	X_{18}	+	X_{27}	+	X_{49}	=	100

Napkin laundry equations:

Day	Napkins sent to Fast Laundry		Napkins sent to Slow Laundry		Napkins sent to Fastest Laundry		Napkins Stored			Napkins Used
	(z)	+	(y)	+	(w)	+	$(s_j - s_{j-1})$			
1	X_{11}	+	X_{21}	+	X_{41}	+	$X_{31}-$	0	=	50
2	X_{12}	+	X_{22}	+	X_{42}	+	$X_{32}-$	X_{31}	=	60
3	X_{13}	+	X_{23}	+	X_{43}	+	$X_{33}-$	X_{32}	=	80
4	X_{14}	+	X_{24}	+	X_{44}	+	$X_{34}-$	X_{33}	=	70
5	X_{15}	+	X_{25}	+	X_{45}	+	$X_{35}-$	X_{34}	=	50
.	.		.		.		.	.		.
.	.		.		.		.	.		.
.	.		.		.		.	.		.
.	.		.		.		.	.		.
10	X_{20}	+	X_{30}	+	X_{50}	+	$X_{40}-$	X_{39}	=	100

The new objective function is, then:

$$0.10(X_1 + \ldots + X_{10}) + 0.03(X_{11} + \ldots + X_{20}) + 0.01(X_{21} + \ldots + X_{30})$$
$$+ 0.04(X_{41} + \ldots + X_{50}) = Z \text{ (Min)}$$

The solution of this new problem is shown in table (4.8), below:

TABLE 4.8

SOLUTION OF THE WARDROOM STEWARD PROBLEM
(MODIFICATION NO. 1)

DAY	NAPKINS PURCHASED	CLEAN NAPKINS RETURNED FROM FAST LAUNDRY	CLEAN NAPKINS RETURNED FROM SLOW LAUNDRY	CLEAN NAPKINS RETURNED FROM FASTEST LAUNDRY	NAPKINS USED DURING DAY	SOILED NAPKINS SENT TO FAST LAUNDRY	SOILED NAPKINS SENT TO SLOW LAUNDRY	SOILED NAPKINS SENT TO FASTEST LAUNDRY	SOILED NAPKINS STORED
1	50				50		50		
2	60				60	10	50		
3	80				80		60		20
4		10	50		70		90		
5			50		50		50		
6			60		60	10	50		
7			90		90		70	20	
8		10	50	20	80	30			50
9			50		50				50
10		30	70		100				100

TOTAL COST = $26.50

It can be seen that in this new problem only once does the Chief Steward use the one day service, and by doing so he saves 20¢ from the previous program.

It should be noted that, as in extending the problem in time, it could just as well be extended also in laundry service sources by simply adding more variables. The limiting factor would be, as before, the computational capacity of the solution equipment being used.

Since this problem has been cast in a somewhat facetious setting, one might question the validity of using linear programming and a computer to determine how best to send napkins to a laundry and thereby save a few pennies. It bears repeating that the problem just solved has much more serious implications, as mentioned previously (such as in repair and maintenance programs). Where we are saving only pennies here, by simply displacing decimal points of the costs associated with the several variables, it may be that millions of dollars could be saved.

Finally, one other point also bears repeating here. As pointed out in chapter II, the context of a problem is not so very important; the structure of the mathematical model is, however, all important. It has been shown how problem settings may change and the mathematical model remains constant; the point being that one must not become steeped and imobilized by classical sterotypes.

CHAPTER V

SPECIAL FEATURES OF LINEAR PROGRAMMING

General

There are two special features of linear programming methods which add to the usefulness and versatility of a linear programming approach to problems. These two facets of the technique are the use of linear programming in (a) sensitivity analysis, and (b) parametric programming studies.

It has been mentioned that one aspect of data, generated within a business organization, is that the data are not always completely accurate, that some data must be estimated, and that some processes must be assumed.

A mathematical model cannot discriminate among data which are accurate to ten significant figures (for example) and data which is not necessarily accurate to even one significant figure. The solution of a model, therefore, will naturally reflect only the data used, and in the numerical examples given thus far it has been assumed, although not specified, that data used was exactly accurate.

By the use of sensitivity analysis and parametric programming one may knowingly use inaccurate data, when necessary, and explore the reactions of a mathematical model under various assumptions.

It must be admitted that the two terms: "sensitivity analysis," and "parametric programming," are almost synonymous terms and there is some confusion in the literature regarding the actual scope intended in this area.

Sensitivity Analysis

Garvin is the only authority, which has been found, to treat the particular subject of "sensitivity analysis."[55] If this concept is considered at all, other authorities generally include it in the subject of "parametric programming."

Garvin[56] describes sensitivity analysis as being applicable, after an optimum solution of a mathematical model has been achieved, and that the concept then encompasses the broad area of five categories, as follows:

a. Changes in the right hand side of the constraining equations (changes in b_i, of the general linear programming statement). This includes variations of the restrictions on various, or specific, resources.

b. Changes in the objective function (changes in c_j). These changes include variations in costs or profit margins.

---

[55] Garvin. Op. Cit., chapter 4.

[56] Ibid., p. 49.

c. Changes in the variable coefficients (changes in a_{ij}). These changes are relative to the variation in usage of resources.

d. Addition of new variables. This includes changes to the product mix and addition or deletion of alternative choices in the decision making process.

e. Addition of constraining equations. This includes variations of product dependency on various resources or additional usage of different resources.

Garvin presents the theoretical approach to sensitivity analysis and how it may be accomplished in hand computations.[57]

The five phase computer program (Linear Programming II [10.1.008]), used in the solutions already presented, may be utilized to accomplish (a) and (b) above, (i.e., changes in the right hand side of the constraining equations, and changes in the objective function costs, or profit). In order to accomplish (c), (d), or (e) above, using the computer program, one must solve an entirely new problem from the beginning; and in fact, this was done in chapter IV (see the "Distribution" and "Wardroom Steward" problems).

---

[57] Ibid., pp. 50-61.

Parametric Programming

Parametric linear programming may be thought of as being part of sensitivity analysis and may be defined as the treatment of some element of the mathematical model (other than X_j) as a parameter of the model. The element, now a parameter, is allowed to vary continuously through some range of values, while the reaction of the system is observed (thus, a parametric study of the system is made).

Garvin limits his definition of parametric linear programming to only a treatment of the right hand side of the constraining equations (b_i) as a parameter.[58] Saaty[59] and Gass[60] discuss only changes in the cost coefficients (c_{ij}) of the objective function. It would seem that the difference among the authorities, as to what parametric programming really is, does not warrant argument.

Example of Sensitivity Analysis and Parametric Programming

Programming, in a strict definition, means planning, and, actually, linear programming finds its most useful application as a planning device. This idea will be

---

[58] *Ibid.*, pp. 220-221.

[59] Saaty. *Op. Cit.*, p. 194.

[60] Gass. *Op. Cit.*, p. 109.

extended somewhat in the following chapter in connection with "long-range planning,"; however, now, consider a simple example of a planning problem.

(Planning Model No. 1)

An oil company, which produces crude oil in two fields and has a system of pipelines, refineries and sales offices to distribute its products to various customers, might have a system as shown in figure (5.1).

Various company officials have submitted data, as their estimates for the following year's operational capabilities and prospects; these data are shown in figure (5.2).

After constructing a proper mathematical model for the data, as originally submitted, and solving the first problem, an optimal solution is obtained as shown in table (5.1A).

The planning committee, being in doubt about the accuracy of the operational costs of refinery "A", decide to investigate the reaction of the mathematical planning model by increasing the operational costs of the refinery, in various increments. It is discovered that the optimal solution remains fixed, although income changes (as shown in figure [5.3]), until the operational costs of refinery "A" have increased a total of 16¢ per barrel (i.e., from a

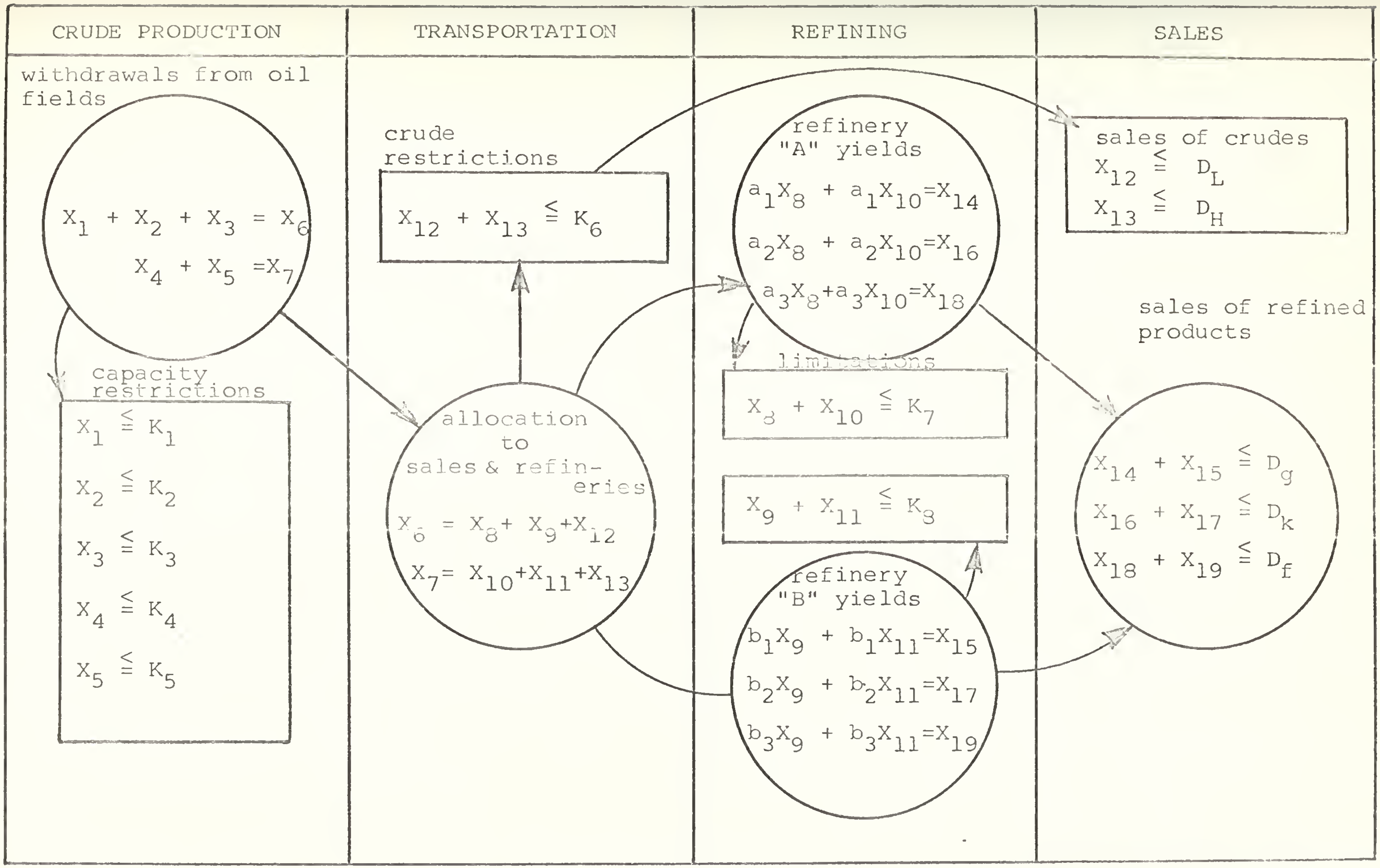
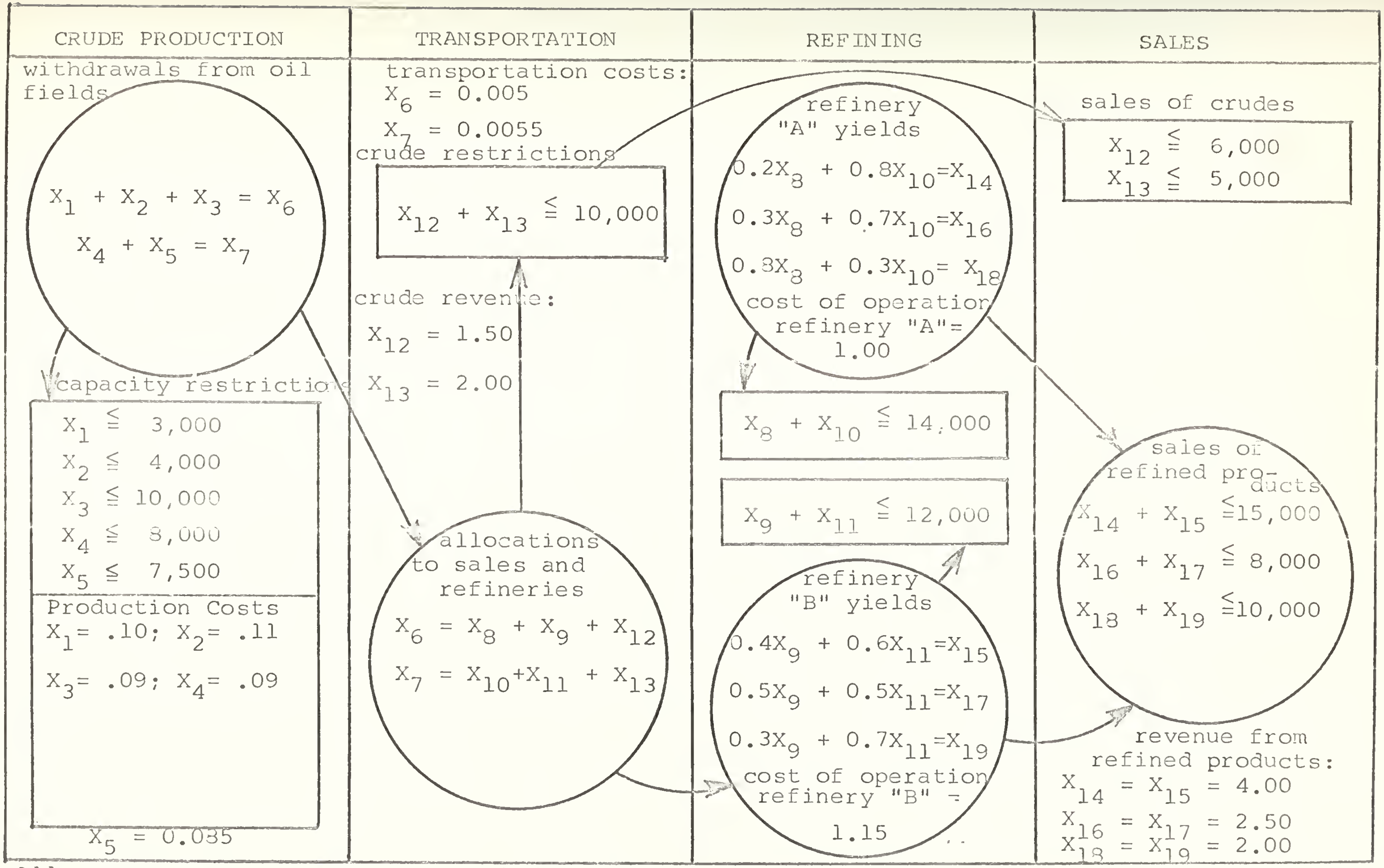


FIGURE 5.1

SCHEMATIC DIAGRAM OF PLANNING MODEL NO. 1



All costs represent cost per bbl ($/bbl.)
All revenues represent revenue per bbl ($/bbl)

FIGURE 5.2
OPERATIONAL DATA OF PLANNING MODEL NO. 1

TABLE 5.1

SOLUTION OF PLANNING MODEL NO. 1

Description		Variable	A Optimum Quantity	B Optimum Quantity
Crude production from oil field No. 1		1	887.0	887.0
2		2	0.0	0.0
3		3	10,000.0	10,000.0
4		4	8,000.0	8,000.0
5		5	7,500.0	7,500.0
Transportation in pipeline No. 1		6	10,887.0	10,887.0
2		7	15,500.0	15,500.0
Refinery usage of produced crude (crude)	(refinery)			
(light)	(A)	8	4,887.0	2,677.0
(heavy)	(A)	10	3,919.0	1,710.0
(light)	(B)	9	0.0	2,210.0
(heavy)	(B)	11	7,581.0	9,790.0
Crude sales				
(light)		12	6,000.0	6,000.0
(heavy)		13	4,000.0	4,000.0
Sales of refined products	(refinery)			
gasoline	(A)	14	4,113.0	1,903.0
gasoline	(B)	15	4,548.0	6,758.0
kerosene	(A)	16	4,210.0	3,000.0
kerosene	(B)	17	3,790.0	6,000.0
fuel oil	(A)	18	4,694.0	2,484.0
fuel oil	(B)	19	5,306.0	7,516.0

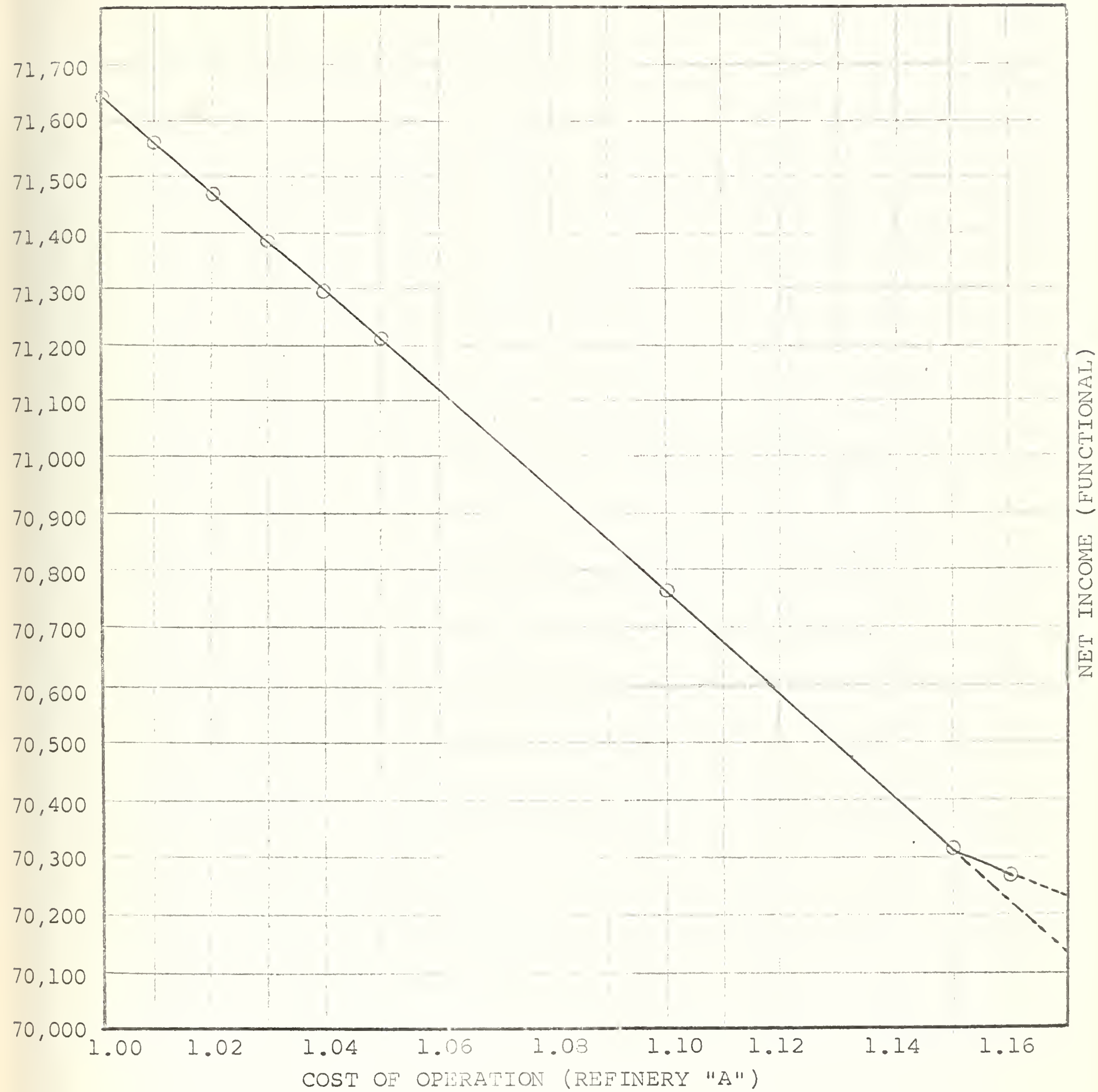


FIGURE 5.3

COST CHANGE RELATIONSHIPS OF REFINERY "A"
(PLANNING MODEL NO. 1)

total of \$1.00/bbl. to \$1.16/bbl.). When this latter condition occurs the optimum solution changes as shown in table (5.1B). These findings are reasonable since, at \$1.16/bbl. operating cost in refinery "A", it is less expensive to allow refinery "B" (with an operating cost of \$1.15/bbl.) to produce more products. Therefore, a shift of crude oil usage is made between refinery "A" and refinery "B" with resulting changes in their respective production of refined products. An exact balance is maintained, however; the crude taken away from refinery "A" exactly equals the increased crude given to refinery "B", and the production lost at refinery "A" is made up at refinery "B".

It can be noted that the market forecasts indicate that a total of 33,000 bbls. of refined products can be sold; however, it is found to be more profitable not to attempt to meet all of the market demands for gasoline (even though the revenue is greatest from gasoline.) Instead, the optimum solution calls for letting oil field No. 2 lie fallow, producing nothing, thereby leaving 4,000 bbls. of crude raw material unused. The reason for this, of course, is that the demands for kerosene and fuel oil have been met (with the optimal solution); to produce gasoline in sufficient quantities to meet its market demand would cause kerosene and fuel oil, which are produced as

"by products" in the refining process, to be unsold. It is wiser, then, not to use the crude oil of field No. 2, than it would be not to sell some kerosene and fuel oil. At this point the planning committee might institute further parametric studies to determine if additional costs should be incurred to develop additional sales for kerosene and fuel oil, in order that all available resources could be profitably used.

The cost changes indicated above were accomplished by the use of the "cost changer phase" of the computer program and were made without reworking a completely new problem. Similarly, changes to the right hand side of the constraining equations may be accomplished by means of the "right hand side changer phase" of the computer program.

By using techniques such as illustrated above a business enterprise may discover important reactions of its system, which otherwise would not be obvious.

The planning model, sensitivity analysis and parametric linear programming are not limited to business enterprises, obviously. These methods could, just as well, be applied to a supply or logistics system (large or small). For example, the following analogy could be drawn:

OIL COMPANY	SUPPLY SYSTEM
oil field.	.equipment/personnel
pipe line.	.procurements
refinery	.depot (warehouse) stock)
sales.	.requisitions

The objective to be optimized, in a logistics system, might be the minimizing of costs; however, one could, also, optimize (maximize) the number of requisitions successfully filled. After reaching one original optimum solution, the techniques described in this chapter could be utilized to study various reactions that a particular logistics system might have under varying system changes.

CHAPTER VI

LINEAR PROGRAMMING AND LONG-RANGE PLANNING

General

The use of linear programming as a planning aid, or technique, was touched upon in chapter V. It is the intent of this chapter to extend that idea somewhat.

In many situations, occurring in business, as well as in U. S. Navy Supply operations, one is not always looking for the one answer to a problem. Generally, what is actually being sought, or what is really needed, is a range of alternatives, or plans, which can be followed in the future when various circumstances are encountered.

In previous chapters, specific numerical examples of linear programming applications have been demonstrated, and certainly linear programming has, in recent years, enjoyed a rapid rise to fame based on special applications; undoubtedly there will be many other specific uses made of this technique in the future. However, it appears that one of the most useful and significant applications to which linear programming may be put, now, and in the future, is in an overall treatment of a system as a composite of interdependent functions.

Long-Range Planning

Although it is not the purpose, here, to fully and adequately discuss long-range planning as a subject in itself, nevertheless, a short resume of the meaning and scope of long-range planning should be profitable, as a background.

Payne describes long-range planning as ". . .one of the really new techniques left to management that gives a company a major competitive advantage."[61] Wrapp defines long-range planning as". . .that activity in a company which sets long-term goals for the firm and then proceeds to formulate specific plans for attaing these goals."[62]

To produce a long-range plan every facet of a business must be explored (insofar as possible) for five, ten, or fifteen years in the future, or for whatever length of time is necessary to carry out company objectives. The long-range plan in final form will tell management: (1) what the company is going to do, (2) how the company should proceed to accomplish the objectives, and (3) when the company should take various actions to accomplish the objectives.

---

[61] Payne, Bruce, "Steps in Long-Range Planning," Harvard Business Review, (March - April, 1957), p. 95.

[62] Wrapp, Edward H., "Organization for Long-Range Planning," Harvard Business Review, (January - February, 1957), p. 38.

There are many aspects to long-range planning which must be taken into consideration by an organization desiring to establish an effective plan. Some of the important facets of long-range planning include:

a. Management policy direction. Top management has the responsibility and cannot disregard its obligation for the establishment of company goals, objectives and policies. Not only in this regard but also in all phases of long-range planning top management support is essential for success.

b. Personnel orientation. Because long-range planning must be comprehensive and tends to "look under all corners of the rug" all company personnel must be educated to understand the reasons for planning in order to prevent prejudicial objections. For a finalized plan to be created and become effective, long-range planning must be a "team" effort and all personnel must understand (insofar as certain sensitivity restrictions or the confidential nature of plans will allow):

(1) The plan itself, and what it is supposed to accomplish.

(2) The "why" of all the steps and phases of the plan.

c. Analysis. To design a workable, effective long-range plan a complete and thorough analysis must be made of:

(1) Present products.

(2) Future sales potentials.

(3) Market trends and marketing variables.

(4) Industry growth.

(5) Company strengths and weakness, manpower, finance, production limitations, etc.

The planner has to learn the facts by identifying:

(1) The opportunities and true costs of products.

(2) The potential contributions of different intra-company operations.

(3) The economically significant cost centers.

Once the facts are known the long-range planner has to know how resources must be allocated to attain the results and goals anticipated; thus he must determine:

(1) How resources are presently allocated.

(2) How resources should be allocated in the future to support activities of greatest opportunity.

(3) What steps are necessary for the company to take to go from present circumstances to what ought to be the circumstances for fulfillment of the anticipated goals.

d. Timing. Once a long-range plan has been designed it must be put into a time frame which is:

(1) A realistic framework within which the company can effectively work.

(2) A flexible framework which can be modified as exigencies occur.

This brief outline of the ramifications of long-range planning is by no means complete but it should serve to show the extent to which the elements of an organization must be examined; in short, a company must search out and analyze all variables pertinent to its future existence.

Planning, of course, is not unique to competitive business enterprises and, as analogies were drawn in the previous chapter, similar comparisons can be made between commerical operations and military supply endeavors.

By examining the brief outline of long-range planning, given above, it should be evident that the real essence of making a long-range plan lies in proper and effective analysis of an organization. Without effective quantitative analysis a long-range plan (if it could be realistically called a plan) would emerge as merely a qualitative description of the futuristic hopes of the organization.

Integration of Planning and Programming

From the perspective of long-range planning and business concepts generally, it is a recognized fact that the future existence of an organization involves the fulfillment of specific or multiple goals (which may not always be the maximizing of profits or minimizing of costs). White[63] presents an interesting essay of possible goals which a firm may have, but, as White states, "from a practical standpoint goals of firms or of individuals typically are expressed in optimal terms."[64] It is not so widely recognized, however, that an organization must "optimize" its operations over the whole of the business system.

The concept of organizational optimization has been derived from the simple fact that different alternatives to a business decision generally (and usually) involve different costs and/or different variables. It is not enough that the quantitative estimates of alternative decisions be balanced one against the other, but the impact of each alternative must be evaluated within the whole of the

---

[63] Boulding, Et al. Op. Cit., pp. 181-201.

[64] Ibid., pp. 186.

system to produce effective long-range plans. The danger to be avoided is "suboptimization," which results when alternatives are considered independently of the system. Suboptimization is a case of optimization for one phase (or element) of an operation, without taking into consideration every factor which has a bearing on the problem, whether in an obvious or in a subtle way.

To overcome the pitfall of suboptimization it is necessary to look simultaneously at all pertinent elements of a system as they progress through time and affect one another. Within its limitations, linear programming offers an ideal approach to long-range planning, while overcoming suboptimization.

To illustrate what may (or must) be done in the realm of long-range planning, the example (planning model No. 1) introduced in chapter V, will be extended. (Figures 6.1 and 6.2 and table 6.1A may be thought of as extensions of figures 5.1 and 5.2 and table 5.1A respectively.)

(Planning Model No. 2)

Assume that the planning committee of the hypothetical company, of the earlier example, wish to look ahead for not only one year but for two years. Due to various contingencies, the company will have accomplished

certain modifications in its operations. The second year's situation is shown in figure 6.1, and the prospective operational and investment data are shown in figure 6.2.

It is convenient when discussing linear programming planning models to distinguish between two types of variables: (1) operational variables and (2) investment variables.

Operational variables are associated with the various operational elements of a business system, and usually are concerned only with the cost of operating the particular business. Investment variables are generally related to the industrial plant and facilities and pertain to capital expenditures. Operational variables describe the operations of a business for a relatively short period of time. Investment variables link operations through time periods (or within a time period) by describing the effect of new and/or different equipment on operations.

In the previous example (planning model No. 1) only operational variables (X_j) were used. In the present example one may notice, in addition to the operational variables, the introduction of investment variables (Y_j). By using investment variables the planning model is given what might be called, and in fact is a new "dimension".

The investment variables are not limited to representing only realities; actually their real usefulness lies in

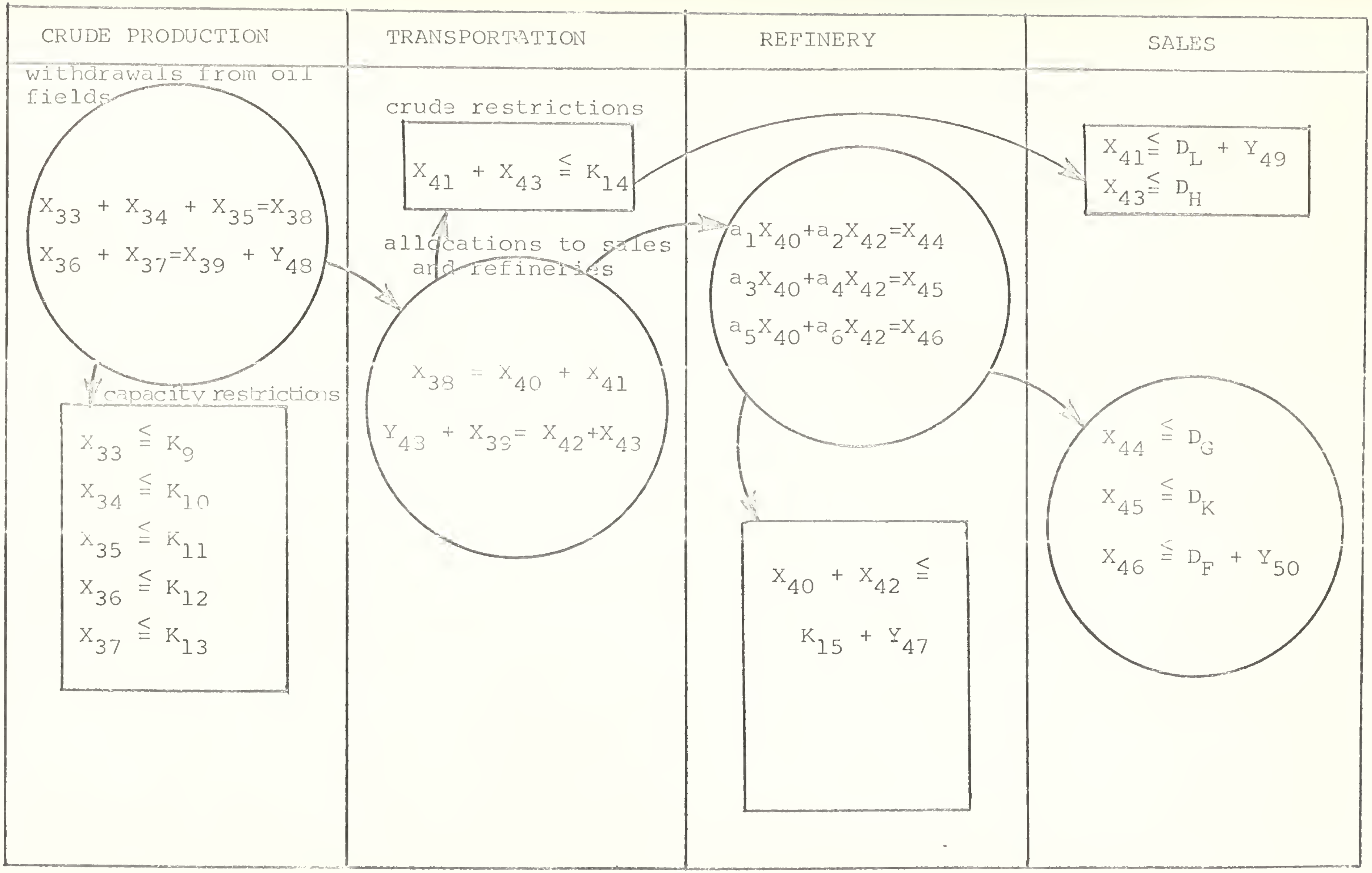
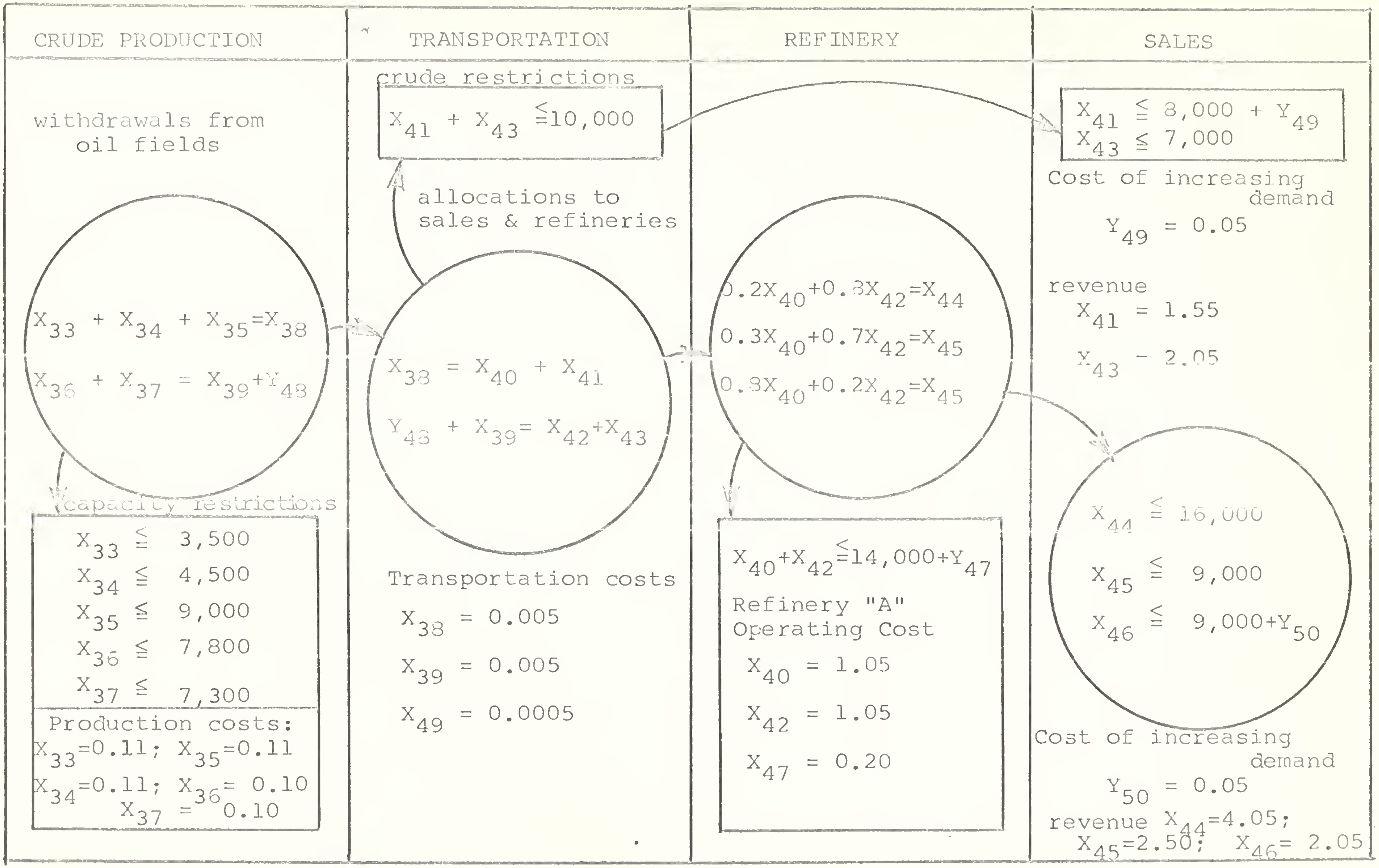


FIGURE 6.1

SCHEMATIC DIAGRAM OF PLANNING MODEL NO. 2



All costs represent cost per bbl.($/bbl.)
All revenues represent revenue per bbl. ($/bbl.)

FIGURE 6.2

OPERATIONAL DATA OF PLANNING MODEL NO. 2

TABLE 6.1

SOLUTION OF PLANNING MODEL NO. 2

Description	Variable	A Optimum Quantity	B Optimum Quantity	C Optimum Quantity
Crude Production from oil field No. 1	X_{33}	3,500.0	833.0	3,500.0
2	X_{34}	4,500.0	4,500.0	4,500.0
3	X_{35}	9,000.0	9,000.0	9,000.0
4	X_{36}	6,657.0	7,800.0	6,300.0
5	X_{37}	7,200.0	7,200.0	7,200.0
Transportation in pipeline No. 1	X_{38}	17,000.0	14,333.0	17,000.0
2	X_{39}	13,857.0	15,000.0	13,500.0
Increment to pipeline #2	Y_{48}	0.0	0.0	0.0
Refinery usage of produced crude in refinery "A" light	X_{40}	14,000.0	11,333.0	14,000.0
heavy	X_{42}	6,857.0	8,000.0	6,500.0
Increment to refinery "A"	Y_{47}	6,857.0	5,333.0	6,500.0
Crude sales light	X_{41}	3,000.0	3,000.0	3,000.0
Increment to sales of light crude	Y_{49}	0.0	0.0	0.0
heavy	X_{43}	7,000.0	7,000.0	7,000.0
Sales of refined products gasoline	X_{44}	8,285.71	8,667.0	8,000.0
kerosene	X_{45}	9,000.00	9,000.00	8,750.0
fuel oil	X_{46}	12,571.0	10,667.0	12,500.0
Increment to sales of fuel oil	Y_{50}	3,571.0	1,667.0	3,500.0

representing proposed system modifications which may, or may not come into being in the future.

In the present example, the planning committee has introduced two investment variables (Y_{49}, Y_{50}) to determine the advisability of increasing the demands for certain products. The investment variables represent the elements necessary for increasing demands, such as: hiring and training of additional salesmen, providing more advertising, increasing product appeal, etc. In table 6.1A it can be seen that the optimal solution calls for expenditures to increase demands for refined fuel oil (Y_{50}) but not for increasing the demands for light crude oil (Y_{49}).

One investment variable (Y_{48}) has been used to investigate increasing the capacity of the pipeline facilities, and one investment variable (Y_{47}) has been used to indicate a possible increase in the processing capacity restriction of refinery "A".

The simplest use for the planning model is determining a solution involving fixed conditions such as in the previous example (planning model No. 1) where the solution may be accepted as being exact. The scope of the planning model, to be really effective, must be extended beyond these limits since conditions are seldom if ever static; however, a word of caution is in order. By the introduction of fictitious elements, or elements which later may be discarded, one has automatically given the

functional a certain spurious nature. In other words, in these situations one cannot, necessarily, accept the "answer" as being literal; the "answer" must be examined with regard to the conditions imposed on the model. In this respect sensitivity analysis and parametric programming may be used to great advantage. For example, the planning committee, using investment variable (Y_{47}) made the initial assumption that, above 14,000 bbls., the capacity of refinery "A" could be increased at a cost of 20¢ per bbl.; by increasing costs to 35¢ per bbl. it is found (as shown in table 6.1B) that the optimal solution changes not only for the investment variable (Y_{47}) but also, among other changes, for investment variable (Y_{50}) representing proposed expenditures for increasing demands for fuel oil. Similarly, by allowing demands for gasoline to decrease from 16,000 bbls. to 8,000 bbls., it is found that there is no change in the optimal solution until demands for gasoline decline to 8,000 bbls. (as shown in table 6.1C). At this latter point the new optimal solution calls for less capacity of the refinery and less expenditure for the development of new demands for refined fuel oil. The interaction and interdependency of the variables, of course, account for changes among the variables which are only indirectly related. The changes in variables (Y_{47}) and (X_{50}), as shown above, prevent

"suboptimization" because the system, rather than individual elements of the system, have been optimized. In addition, it becomes obvious that any one solution is dependent upon many factors and the static conditions prevelent at the time; given some change in fixed conditions the solution may, and probably will, be different.

In addition to investigation of real and hypothetical elements of a system, as indicated above, the planning model may be used for many other planning analyses. However, the planning model cannot describe anything about a great many things such as the criteria on which assumptions were based in order to formulate the model; these criteria must be set by management using experience and judgement. Also, there are a great number of other factors involved in business which are (at present, anyway) non-quantifiable. Such factors include: company "goodwill"; labor relations, public relations, ethical practices, legalities, etc. As Rapoport and Drews have stated "the establishment of criteria as to what is actually best for a business enterprise is altogether beyond the realm of mathematics."[65]

Although in the example used, here, the time span covered has been only two years, it could have been extended

---

[65]Rapoport and Drews. Op. Cit., p. 79.

well beyond that. The limiting factor is, again as we have seen before, the computational capacity of the equipment being used. For example, Rapoport and Drews estimate that a realistic planning model, covering five successive time periods, would contain perhaps 500 variables and about 400 equations."[66] A modest model of that size would require considerable computational capacity, but even with this and other disadvantages, a linear programming approach to long-range planning offers decision makers a very powerful tool for analysis in determining future courses action.

---

[66] Ibid., p. 86.

CHAPTER VII

SOME INVESTIGATIONS INTO EXTENSIONS OF LINEAR PROGRAMMING

General

During research for this thesis an attempt has been made by this author, to develop, to a limited extent, certain new and different approaches to two linear programming problems. This chapter consists of the results of the investigations in this area.

The Loading Problem

The loading problem, which is of some military value if not also of some theoretical interest, has been mentioned by one author as having had no significant theory advanced for its solution (as of 1960).[67]

The loading problem may be stated simply as follows: Suppose some vehicle (an airplane or ship, for example) is to be loaded up to or including a certain maximum weight (W). The cargo which is to be loaded is in the form of discrete packages, or bundles, each having individual values (v_i) and individual weights (w_i). The problem presented, then, is how to load the vehicle with the maximum

---

[67] Gale, David. The Theory of Linear Economic Models. New York: McGraw-Hill Book Co., Inc., 1960. p. 134.

value of cargo but not exceeding the maximum allowable weight load limit (W).

The problem has been stated in linear programming form:[68]

Maximize: $$\sum_{i=1}^{n} E_i v_i$$

$$E_i = 1, \text{ or } E_i = 0$$

subject to: $$\sum_{i=1}^{n} E_i w_i \leqq W$$

Present known methods of solving linear programming problems, such as the simplex method, cannot be used to solve this problem since they do not guarantee integer results. Even the transportation method of solution, which yields integer values to certain types of linear programming problems where the problem may be formulated by means of double description or double identification of the variables, cannot be used. The transportation method of solution requires known source availability values and destination requirement values as well as the various costs associated with moving an item from one location to another. In the loading problem variables cannot be described in analogous terms suitable for submission to the transportation algorithm.

---

[68] Ibid.

The loading problem may be solved, in one way, by direct enumeration. Here one merely examines, in turn, each of the possible sets of objects to be loaded, and among those which do not exceed the weight limit, one chooses those which will yield the greatest total value for the combined cargo.

The difficulty involved in attempting to compare all possible combinations is the magnitude of the number of comparisons required. The total number of combinations of "n" things taken: zero at a time, one at a time, two at a time,. . .up through "n" at a time is 2^n (or $2^n - 1$, if zero at a time is not considered). It is easily seen that the total number of possible combinations rapidly becomes very large and quickly outgrows our time and ability for making comparisons (for example, just ten packages would require 1,024 comparisons to assure finding the optimum loading schedule; 30 packages would require 1,073,741,824 comparisons, etc.). The direct enumeration method, therefore, is not a practical method of solution for the loading problem (although direct enumeration will be used to check the procedure advanced below).

The author now presents the following original heuristic development as a procedure for solution of the loading problem.

Be re-examination of the problem statement it can be seen that the restriction

$$\sum_{i=1}^{n} E_i w_i = W$$

may be restated, after all E_i are set equal to unity, as follows:

$$\sum_{i=1}^{n} w_i - S = W$$

Where "S" is some slack amount which cannot be loaded. The problem, then, could be restated by saying that we desire to minimize "S". We have, as yet, said nothing about the values (v_i) of the individual weights (w_i) and we must have the two variables related in some way. If we introduce an additional variable "X", which shall be defined:

$$X_i = v_i/w_i \quad . \; . \; . \; . \; . \; . \; . \; . \quad (7.1)$$

$$w_i = v_i/X_i \quad . \; . \; . \; . \; . \; . \; . \; . \quad (7.2)$$

and which will be called, simply, an index, we will then have a connection between weights and values.

Using equation (7.2) the problem may, again, be restated as follows:

Minimize: S

where: $$\sum_{i=1}^{n} v_i/X_i - S = W$$

$$-S = W - \sum_{i=1}^{n} v_i/X_i$$

$$S = -W + \sum_{i=1}^{n} v_i/X_i$$

Then, for "S" to be a minimum the derivative of "S" with respect to each individual value (v_i), (dS/dv_i), must approach zero. Taking the derivative we obtain:

$$dS/dv_i = \frac{d(-W + \sum_{i=1}^{n} v_i/X_i)}{dv_i}$$

then:

$$dS/dv_i = d\,\frac{(\sum_{i=1}^{n} v_i/X_i)}{dv_i}$$

or:

$$dS/dv_i = \frac{d}{dv_i}\,(v_1/X_1 + v_2/X_2 + v_3/X_3 + \ldots + v_n/X_n)$$

then:

$$dS/dv_i = 1/X_i$$

For the derivative (dS/dv_i) to approach zero, each individual index (X_i) must be as large as possible. In other words, by choosing bundles having the largest indexes we should, theoretically, have a loading schedule containing the greatest value of cargo with the minimum amount of weight.

If, then, we are given a listing of various weights and their associated values, by dividing each value by its respective weight we may obtain an index. Arranging the various indexes in descending order of magnitude in a listing with the respective weights and values we may choose, from the beginning of the list, those weights which add to, or are just less than the desired maximum load limit. For example, if we were given the following weights and values:

WEIGHT	VALUE
10.	10.
1.	1.
5.	10.
2.	10.

a new listing, arranged by descending index, could be formed, as follows:

PACKAGE	WEIGHT	VALUE	INDEX = v/w
(a)	2.	10.	5.0
(b)	5.	10.	2.0
(c)	1.	1.	1.0
	10.	10.	1.0

Then, if we were given various loading objectives, loading schedules containing the optimum load could be prepared, as follows:

MAXIMUM WEIGHT (LOAD OBJECTIVE)	LOADING SCHEDULE (PACKAGES)	LOADING WEIGHT (OPTIMUM WT.)	CARGO VALUE
2	(a)	2	10.
3	(a), (c)	3	11.
4	(a), (c)	3	11.
5	(a), (c)	3	11.
7	(a), (b)	7	20.
10	(a), (b), (c)	8	21.

It should be noted that where indexes are identical the smalles weight should precede the larger weight in the list (as between packages (c) and (d)); where a weight exceeds the remaining allowable load it must be skipped and

the remaining weights chosen which are equal to or less than the remaining allowable load (as shown in obtaining the loading schedule for weight objectives 3, 4, and 5).

At this point one has obtained a loading schedule which is equal to or less than the maximum allowable load limit and which has what might be called the highest "value density" of all possible loads. However, since we must find the highest absolute value of the cargo within the allowable range, the load having the highest "value density" will not necessarily be the loading schedule having the highest total absolute value. This may be explained by referring to the initial discussion concerning the minimizing of the slack (S). By taking the derivative (dS/dv_i) we have tacitly understood that we have an infinite number of bundles, or packages, of infinitesimally small incremental weight and values. Such, unfortunately, is not the case, as a practical matter; if it were we would have already solved the loading problem.

Once, having obtained the load of highest "value density" it may be possible to improve on this loading schedule by replacing some weight or weights on the schedule by some weight or weights not included in the loading schedule.

The algorithm may be followed, most easily, in a step-by-step process, as follows:

Step 1. An index (X_i) is computed for all data points, which consist of various weights (w_i) and associated values (v_i):

$$X_i = v_i/w_i$$

Step 2. All data are then sorted into a list by descending order of index.

Step 3. The weight (w_i) of the first data point is compared with the maximum allowable load limit (WMAX). There are three possible courses of action:

a. If: w_i < WMAX, then the data point (w_i, v_i) is put into a list (to be called the "loading schedule"). One would then go to step 4.

b. If: w_i > WMAX, then the data point (w_i, v_i) is put into a list (to be called the "excluded list"). One then would go to step 4.

c. If: w_i = WMAX, then the data point (w_i, v_i) is placed in the loading schedule and all other remaining data points are placed in the excluded list in descending order of index. One then would go to step 5.

Step 4. The quantity (WMAX) is reduced by the amount of the weight (w_i) just placed in the loading schedule, so that WMAX now equals the balance which can be loaded. One now repeats step 3, using, in order, the second, third, fourth, etc., data points until all data have been exhausted in the comparison with WMAX.

At this point one has obtained a first feasible solution. A checking procedure is next carried out to determine if this first feasible solution may be improved.

Step 5. The data points (w_i, v_i) of the loading schedule are sorted into a listing which is in order by ascending values, (v_i), where v_1 is the smallest value on the loading schedule and v_n is the largest value).

For simplicity we will denote the data points on the loading schedule by the additional subscript (s) such as ($w_{s,\ i}$, $v_{s,\ i}$) and we will denote the data points on the excluded list by the additional subscript (o), such as ($w_{o,\ j}$, $v_{o,\ j}$). If, then, "n" equals the number of data points on the loading schedule, and "N" equals the number of data points on the excluded list, then: (i=1,2,. . .,n) and (j=1,2,. . .,N).

Step 6. The value ($v_{o,\ j}$) where j=1 for the first data point on the excluded list is compared with the value ($v_{s,\ i}$) where i=1 for the first data point on the loading schedule, and letting: $v_{s,\ i}$ = VS, and $w_{s,\ i}$ + S = WS, where "S" equals the difference between the maximum allowable load limit and the total weight on the loading schedule, then the alternatives are:

a. If: $v_{o,\ j} \leqq$ VS, then no replacement can be made, and one would go to step 9.

b. If: $v_{o,\ j} > VS$, a replacement may be possible, and one would go to step 7.

Step 7.

a. If: $w_{o,\ j} \leq WS$, then $(w_{o,\ j})$ and $(v_{o,\ j})$ are placed at the end of the loading schedule (the n+1 position). After the data point is placed at the end of the loading schedule, a check is made to determine if any of the data points which make up WS can remain on the loading schedule. Then, those data points included in WS which have to be removed from the loading schedule are set equal to zero. One then would go to step 9.

b. If: $w_{o,\ j} > WS$, then we make further comparisons; one would go to step 8.

Step 8.

a. If the value of the next data point $(v_{s,\ i+1})$ is less than the value of the excluded point being considered:

$v_{s,\ i+1} < v_{o,\ j}$, then one would add the data point $(w_{s,\ i+1}, v_{s,\ i+1})$, on the loading schedule, to WS and VS respectively, so that:

$$WS_{new} = WS_{old} + w_{s,\ i+1}$$

$$VS_{new} = VS_{old} + v_{s,\ i+1}$$

Then, one would follow either step 6a or step 6b, as appropriate.

b. If: $v_{s,\ i+1} \geqq v_{o,\ j}$, then one would take the value of the next data point and repeat step 8.

c. If: step 8 is reached and all data points on the loading schedule have been considered in comparison with the data point ($w_{o,\ j}$, $v_{o,\ j}$) on the excluded list, such that $w_{o,\ j} > WS$, then no exchange between the two lists is made and one would go to step 9.

Step 9. The next data point ($w_{o,j+1}$, $v_{o,\ j+1}$) in the excluded list is then used in the same way as point $w_{o,j}$, $v_{o,j}$ and one then would follow step 6 from the beginning. This procedure is followed until all of the data in the excluded list have been checked.

Step 10. After the above procedure is followed, the answer has been obtained, as shown by the loading schedule.

A computer program (written in Fortran II language) was developed in an attempt to solve the loading problem, as presented above (for simplicity this program will be referred to as the approximation algorithm). In addition four other computer programs were prepared in conjunction with attempting to determine the accuracy of the approximation algorithm. These latter four programs consist of:

a. A random data generator which prepares simulated weight and value data and simulated maximum load limits.

b. A direct enumeration program which makes all possible comparisons for six weights and six values (64 computations).

c. A direct enumeration program which makes all possible comparisons for ten weights and ten values (1,024 computations).

d. A direct enumeration program which makes all possible comparisons for twelve weights and twelve values (4,096 computations).

Copies of all the above mentioned programs may be found in Appendix I.

The simulated data, prepared from the random data generator program, was submitted first to the appropriate direct enumeration program to determine the correct answer and then to the approximation algorithm to determine its accuracy. The results of these tests are shown below in table 7.1 and table 7.2.

TABLE 7.1

RESULTS OF TESTS CONDUCTED WITH APPROXIMATION ALGORITHM

NUMBER OF DATA POINTS	TOTAL NUMBER OF TEST SETS	INCORRECT ANSWERS FROM APPROXIMATION ALGORITHM	PER CENT ERROR
6	38	4	10.5
10	33	6	18.2
12	10	1	10.0
TOTAL	81	11	13.6

TABLE 7.2

ANALYSIS OF ERRORS MADE BY APPROXIMATION ALGORITHM IN EIGHTY-ONE TESTS

Weight Results						
No.	Number of Data Points	Maximum Load Limit	Direct Enumeration Weight	Weight by Approximation Algorithm	Weight Difference	Per Cent
1	12	1,101.5	1,000.0	872.0	-128.0	-12.8
2	10	365.9	679.0	573.0	-142.0	-20.9
3	10	919.1	838.0	671.0	-167.0	-20.0
4	10	569.2	533.0	504.0	- 29.0	- 5.4
5	10	1,662.6	1,662.0	1,349.0	-313.0	-18.8
6	10	3,514.3	3,456.0	3,254.0	-202.0	- 5.8
7	10	407.6	392.0	292.0	-100.0	-25.5
8	6	1,455.6	1,337.0	1,378.0	+151.0*	+12.3*
9	6	1,331.9	1,075.0	1,077.0	+ 2.0*	+ 0.2*
10	6	1,437.6	1,227.0	904.0	-223.0	-18.2
11	6	779.4	750.0	342.0	-408.0	-54.5

Value Results						
No.	Number of Data Points		Direct Enumeration Value	Value by Approximation Algorithm	Value Differences	Per Cent
1	12		2,932.0	2,895.0	- 37	- 1.26
2	10		1,646.0	1,457.0	-189	- 8.7
3	10		1,981.0	1,940.0	- 41	- 2.6
4	10		2,396.0	2,378.0	- 18	- 0.75
5	10		2,817.0	2,496.0	-321	-11.5
6	10		3,102.0	3,089.0	- 13	- 0.42
7	10		1,986.0	1,875.0	-111	- 5.6
8	6		2,200.0	1,645.0	-555	-25.2
9	6		2,935.0	2,623.0	-312	-10.6
10	6		2,200.0	2,129.0	- 71	- 3.2
11	6		1,118.0	980.0	-138	-12.4

*Approximation algorithm weight exceeded optimum weight determined by direct enumeration.

Example solutions of five sample problems, together with direct enumeration computations, may be found in Appendix II.

It should be pointed out that data which were used in the above tests were of a realistic nature; no data were included in the tests which would have resulted in "no contest".

Although these tests are not entirely conclusive it appears that the approximation algorithm tends toward being exactly accurate approximately 80% of the time and approximately 20% of the time it would be within approximately 75% of the maximum possible loading value. Depending, of course, on the accuracy which is required in determining the maximum possible value of any given cargo, it would seem that the loading approximation algorithm would be sufficiently accurate. The worth of the approximation algorithm is enhanced when computing times (utilizing an IBM 1620 computer) are considered: (see table 7.3, below).

TABLE 7.3

COMPARATIVE SOLUTION TIMES FOR LOADING PROBLEMS

Number of Data Points	Total No. of Data Sets	Computation Time per Set by Direct Enumeration (t_c)	Total Time by Direct Enumeration	Total time by approx. algorithm for total data sets
6	38	11 sec.	7 min.	6 min.
10	33	3.5 min.	2 hours	5.5 min.
12	10	15 min.	2½ hours	2 min.

Although computing times increase with the approximation algorithm, as the number of items being considered increases; however, the increase per data point is on the order of micro-seconds. On the other hand, by direct enumeration, which, of course, will give an absolutely correct answer, computing times are slightly more than doubled with each additional item, or data point, considered (see figure 7.0).

For example, to compare twelve items the computing time is approximately 15 minutes; to compare thirteen items the computing time would be approximately 30 minutes. It is interesting to note that, by direct enumeration, with constant computation, the time required for comparing only thirty-one items would be in excess of fifteen years, using an IBM 1620 computer. In figure 7.0 it can be seen that

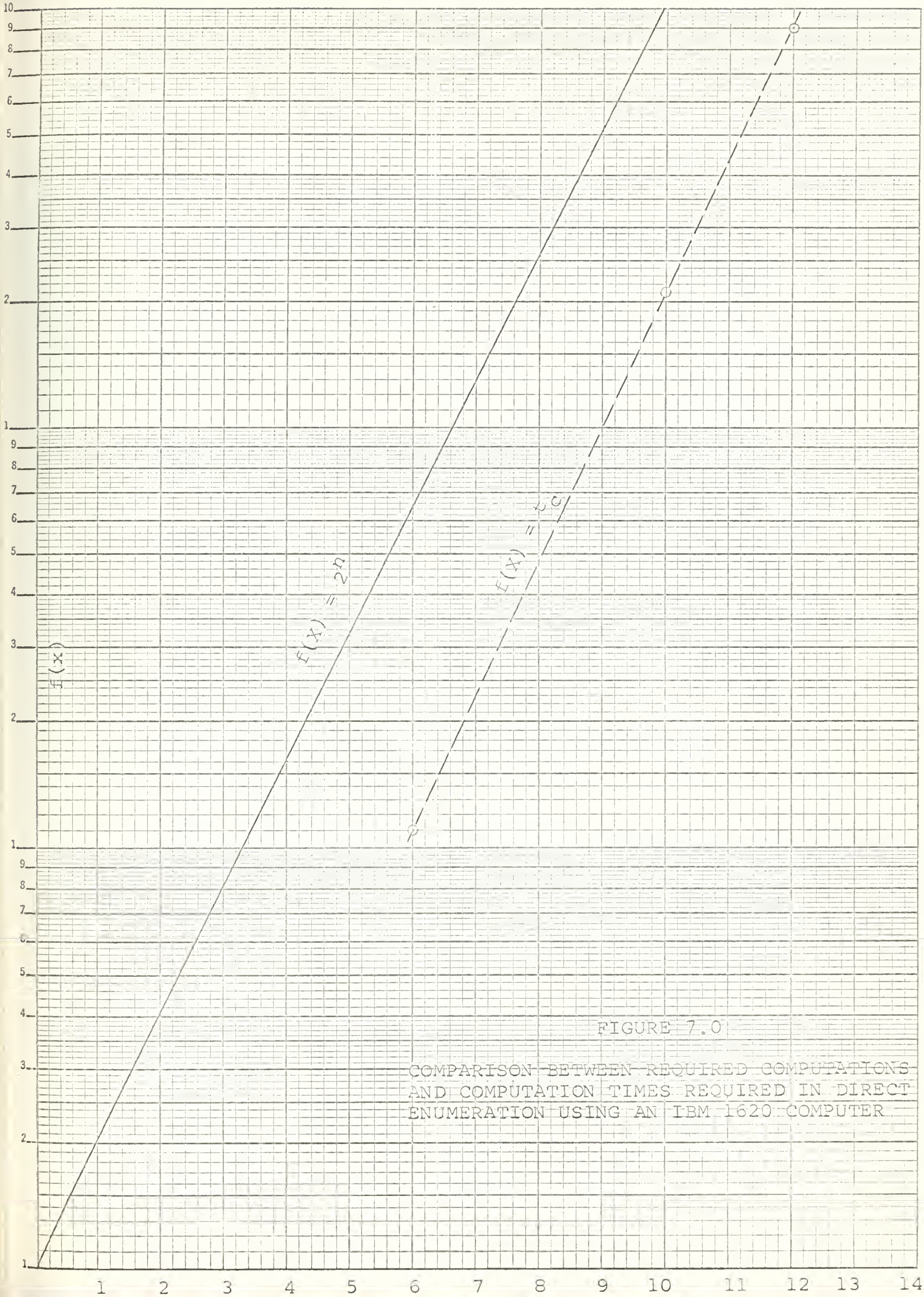


FIGURE 7.0

COMPARISON BETWEEN REQUIRED COMPUTATIONS AND COMPUTATION TIMES REQUIRED IN DIRECT ENUMERATION USING AN IBM 1620 COMPUTER

the computation time increase at a somewhat greater rate than the number of comparisons being made.

The loading problem offers many interesting aspects, and during research for this thesis, it has been approached from several points of view, most of which proved fruitless.

It is believed, although this author cannot prove his belief, that the loading problem has no direct solution (other than the direct enumeration method). The author has been led to this belief after considering that the data with which one deals in the loading problem, must be considered as being completely random, and, therefore, any one direct method which would solve the problem in one case would, most probably, fail in another case when the data assumed a different distribution pattern. In addition, there are an infinite number of combinations (when decimals are allowed) which can combine to make some given total, and any number (say, X_1) which is in the set of data and which is less than the maximum allowable load (say, M) belongs to at least two sets of numbers. The number (X_1) belongs to both an infinite and a finite set of numbers which exactly totals to the objective (M); but some of the other numbers in a common set with (X_1) are not in the available data. The number (X_1) also belongs to a finite and an infinite set of numbers which totals to some value less than the objective (M), and the other numbers of this common set may or may not be in the available data.

The other numbers in the list of data (say, X_2, X_3,. . .,X_n) which is available belong to similar sets of numbers as does X_1. Then, essentially the loading problem is one of finding among all the available data, a common finite set of numbers which either sums exactly to (M):

$$\sum_{i=1}^{n} X_i = M$$

or a common finite set of numbers, from the available data, which sums exactly to some number less than (M):

$$\sum_{i=1}^{n} X_i = M - S$$

Therefore, what one has to determine is one finite set of numbers from an infinite number of sets, without knowing what set of numbers one is looking for. In essence this is analogous to, but worse than, "looking for a needle in a haystack".

The author believes, and here again he cannot prove this, that the loading problem may eventually be solved by some trial and error procedure. One procedure, which was not attempted, but which would seem to have good possibilities for almost complete accuracy, is a method utilizing sampling theory. A computer program could be developed to compute (within the desired accuracy) the number of samples one would require and then proceed to take

random samples of the data for the required number of times, or for some very large number of times, with the best answer, of course, being stored while all others would be discarded. It would seem, also, that a sampling approach could be applied to other intractable problems as well.

<u>The Loading Problem Including Cube (Volume) Considerations</u>

During the course of research and study of the loading problem the idea occurred that it might be extended to include cube (or volume) considerations in addition to just merely weight restriction.

In this new problem, as in the preceding one, imagine that a vehicle (such as a ship or airplane) is to be loaded up to or including a certain maximum weight (W) but also not exceeding a maximum allowable volume, or cube (C). Again, the cargo which is to be loaded is in the form of discrete packages, or bundles, each having individual values (v_i), individual weights (w_i) and individual volumes, or cube (c_i).

The problem presented is how to load the vehicle with the maximum value of cargo but not exceeding the maximum allowable weight load limit (W) and also not exceeding the maximum allowable volume or cube limit (C).

Although not found elsewhere specified as a linear programming problem, this problem will, here, be treated in

the same manner as the preceding loading problem which had only weight limitations.

In linear programming form we might present the problem as follows:

Maximize: $$\sum_{i=1}^{n} E_i \; v_i$$

subject to: $$E_i = 1, \text{ or } \; E_i = 0$$

$$\sum_{i=1}^{n} E_i \; w_i = W$$

$$\sum_{i=1}^{n} E_i \; c_i = C$$

For the same reasons given before this problem cannot be solved by present known methods of solving linear programming problems. Of course, as before, and with the same obvious drawback of too extensive comparisons, the problem might be solved by direct enumeration.

In attempting a solution of this new problem we might begin by restating the restrictions:

$$\sum_{i=1}^{n} E_i \; w_i = W$$

$$\sum_{i=1}^{n} E_i \; c_1 = C$$

after all E_i are set equal to unity, as:

$$\sum_{i=1}^{n} w_i - S = W$$

$$\sum_{i=1}^{n} c_i - S' = C$$

where "S" and "S'" are slack amounts of weight and cube, respectively, which cannot be loaded on the vehicle. Then, if we define a new variable "X", which shall be called an index, as:

$$X_i = v_i/(c_i)(w_i) \quad \dots\dots\dots \quad (7.3)$$

$$w_i = v_i/(c_i)(X_i) \quad \dots\dots\dots \quad (7.4)$$

$$c_i = v_i/(w_i)(X_i) \quad \dots\dots\dots \quad (7.5)$$

we have a relationship among the respective weights, values, and volumes of the several packages.

If we approach this problem as one of minimizing the slack we may then state the problem, using equations (7.4) and (7.5) as follows:

Minimize: S and S'

where:

$$\sum_{i=1}^{n} v_i/(c_i)(X_i) - S = W$$

$$-S = W - \sum_{i=1}^{n} v_i/(c_i)(X_i)$$

$$S = -W + \sum_{i=1}^{n} v_i/(c_i)(X_i)$$

$$\sum_{i=1}^{n} v_i/(w_i)(X_i) - S' = C$$

$$-S' = C - \sum_{i=1}^{n} v_i/(w_i)(X_i)$$

Then for "S" and "S'" to both be a minimum the derivatives of "S" and "S'" with respect to the individual values (v_i), (dS/dv_i) and (dS'/dv_i), must each approach zero. Taking these derivatives we obtain:

$$dS/dv_i = \frac{d}{dv_i}\left[-W + \sum_{i=1}^{n} v_i/(c_i)(X_i) \right]$$

$$dS/dv_i = \frac{d}{dv_i}\left[\sum_{i=1}^{n} v_i/(c_i)(X_i) \right]$$

$$dS/dv_i = \frac{d}{dv_i}\Big[v_1/(c_1)(X_1) + v_2/(c_2)(X_2) + . . .$$

$$+. . v_3/(c_3)(X_3) +. . .$$

$$+. . v_n/(c_n)(X_n) \Big]$$

$$dS/dv_i = 1/(c_i)(X_i)$$

$$dS'/dv_i = \frac{d}{dv_i}\left[-C + \sum_{i=1}^{n} v_i/(w_i)(X_i)\right]$$

$$dS'/dv_i = \frac{d}{dv_i}\left[\sum_{i=1}^{n} v_i/(w_i)(X_i)\right]$$

$$dS'/dv_i = \frac{d}{dv_i}\left[v_1/(w_1)(X_1) + v_2/(w_2)(X_2) +. . +. .+ v_n/(w_n)(X_n)\right]$$

$$dS'/dv_i = 1/(w_i)(X_i)$$

For the derivatives (dS/dv_i) and (dS'/dv_i) to approach zero individual indexes (X_i) must be as large as possible. In other words, in this problem as in the preceding one, by choosing bundles having the largest indexes we should, theoretically, have a loading schedule containing the greatest value of cargo with the minimum amount of weight and the minimum amount of volume.

Without providing a demonstration of a numerical example at this point (numerical examples are given in Appendix II) let it be sufficient to point out that in this problem, a first feasible solution is obtained in the same manner as in the preceding problem. Packages having weights and volumes less than any remaining allowable load weight

and allowable load volume are chosen by index number in descending sequence. The first feasible solution is the load of highest "value density".

For the same reason given in the preceding loading problem the first feasible solution having the highest "value density" is not necesarily the solution having the maximum possible absolute value. It may be possible to improve on this loading schedule by replacing some weight or weights on the schedule by some weight or weights not included in the loading schedule.

The step by step procedure of the loading algorithm with volume considerations is the same as for the previous algorithm except that the index (X_i) is computed, as follows:

$$X_i = v_i/(w_i)(c_i)$$

and additional methods of comparison for (c_i) are used which are the same as were shown for (w_i) in the previous algorithm.

A computer program (written in Fortran II language) was developed to solve this loading problem containing volume restrictions as presented above. The computer program is included in Appendix I. Five numerical examples solved by this algorithm, and by direct enumeration, may be found in Appendix II. Although this algorithm has not

been tested in the manner in which the loading problem was tested, it is believed that similar results would occur as with the previous tests.

Transformation as a Means Toward Optimization

Unfortunately, not all (or even a majority) of relationships encountered in real life situations occur in neat linear form. A portion of the research that has been conducted in linear programming has been devoted to treating some non-linear relationships, by various approximating techniques, as linear relationships (for example, see Charnes and Cooper[69]). A great number of non-linear relationships are of such a nature, however, that they may be readily transformed, in some manner, into linear relationships not requiring approximation. It would seem, therefore, that relationships of this nature could be treated, upon transformation, by standard linear programming techniques when optimation is desired.

Two types of relationships, which are not only very useful but also frequently met in engineering and industrial enterprises, are those described by exponential and hyperbolic curves. These relationships take the general form:

EXPONENTIAL CURVE	HYPERBOLIC CURVE
$y = ae^{-bx}$	$y = ax^{-b}$

[69] Charnes & Cooper. Op. Cit., Chapter X.

These curves upon transformation take the following general forms:

EXPONENTIAL CURVE	HYPERBOLIC CURVE
$\ln y = \ln a - bx$	$\ln y = \ln a - b \ln x$
$\ln y + bx = \ln a$	$\ln y + b \ln x = \ln a$

The exponential curve becomes linear in semi-logarithmic coordinates and the hyperbolic curve is linear in logarithmic coordinates.

In the following two sections simulated maximization problems of the two general types shown above are presented utilizing the simplex algorithm to reach optimum solutions. Simulated problems involving only two variables have been chosen in order to present graphical as well as computer solutions.

Although multi-variable problems have not been considered it would appear that, as long as linear transformation is possible, there should be no prohibition to the use of linear programming techniques to optimization of multi-variable non-linear problems. For example, there does not seem to be any reason why relationships of the following general form could not also be optimized by the simplex linear programming algorithm:

$$y = ax^{-b}\ cw^{-t}\ gz^{-v}$$

transofrmed to:

$$\ln y = \ln a - b \ln x + \ln c - t \ln w + \ln g - v \ln z$$

$$\ln y + b \ln x + t \ln w + v \ln z = \ln a + \ln c + \ln g$$

where:

a, b, c, g, t, and v are some known constants

w, x, y, and z are some unknown variables.

By letting:

$$\ln y = X_1$$

$$\ln x = X_2$$

$$\ln w = X_3$$

$$\ln z = X_4$$

$$\ln a + \ln c + \ln g = K$$

then:

$$X_1 + bX_2 + tX_3 + vX_4 = K$$

Therefore, in the above manner, the standard linear programming problem formulation can be achieved.

<u>Exponential</u> (<u>Semi-Logarithmic</u>) <u>Maximization</u> <u>Problem</u>

Assume the following abstract system:

Objective (Maximize): $z = ye^{0.231x}$ subject to the following restrictions:

$$y \leqq 8e^{-0.4624x} \quad \text{. . . . .} \quad (7.6.1a)$$

$$y \leqq 10e^{-0.384x} \quad \text{. . . . .} \quad (7.6.1b)$$

$$y \leqq 3.35e^{-0.173x} \quad \text{. . . . .} \quad (7.6.1c)$$

$$y \leqq 5e^{-0.230x} \quad \text{. . . . .} \quad (7.6.1d)$$

$$y \leqq 2.0 \quad \text{. . . . .} \quad (7.6.1e)$$

$$y \leqq 6e^{-0.3584x} \quad \text{. . . . .} \quad (7.6.1f)$$

$$y \geqq 0.0$$

$$x \geqq 0.0$$

$$z > 0.0$$

The above system may be transformed by the use of logarithms, into a linear system as follows:

Objective (Maximize): $Z = \ln z = \ln y + 0.231x$

$$\ln y + 0.462x \leqq \ln 8 \quad \text{. . . . .} \quad (7.6.2a)$$

$$\ln y + 0.384x \leqq \ln 10 \quad \text{. . . . .} \quad (7.6.2b)$$

$$\ln y + 0.1734x \leqq \ln 3.35 \quad \text{. . . . .} \quad (7.6.2c)$$

$$\ln y + 0.230x \leqq \ln 5 \quad \text{. . . . .} \quad (7.6.2d)$$

$$\ln y \leqq \ln 2 \quad \text{. . . . .} \quad (7.6.2e)$$

$$\ln y + 0.358x \leqq \ln 6 \quad \text{. . . . .} \quad (7.6.2f)$$

By allowing:

$$\ln y = X_1$$

$$x = X_2$$

the linear system, as plotted on semi-logarithmic coordinate graph paper in figure 7.1, is as follows:

Objective (Maximize): $Z = X_1 + 0.231X_2$

$$X_1 + 0.462X_2 \leqq 2.07944 \quad (7.6.3a)$$

$$X_1 + 0.384X_2 \leqq 2.30259 \quad (7.6.3b)$$

$$X_1 + 0.173X_2 \leqq 1.20896 \quad (7.6.3c)$$

$$X_1 + 0.230X_2 \leqq 1.60944 \quad (7.6.3d)$$

$$X_1 \leqq 0.69315 \quad (7.6.3e)$$

$$X_1 + 0.358X_2 \leqq 1.79176 \quad (7.6.3f)$$

By the introduction of slack variables the inequations are converted to equalities suitable for submission to the simplex algorithm:

$X_1 + 0.231X_2 = Z$ (Maximize) Objective

$$X_1 + 0.462X_2 + X_3 = 2.07944$$

$$X_1 + 0.384X_2 + X_4 = 2.30259$$

$$X_1 + 0.173X_2 + X_5 = 1.20896$$

$$X_1 + 0.230X_2 + X_6 = 1.60944$$

$$X_1 + X_7 = 0.69315$$

$$X_1 + 0.358X_2 + X_8 = 1.79176$$

The above data, in the proper computer program format, is shown in Appendix II along with the computer solution to the proglem. The computer output indicates the solution

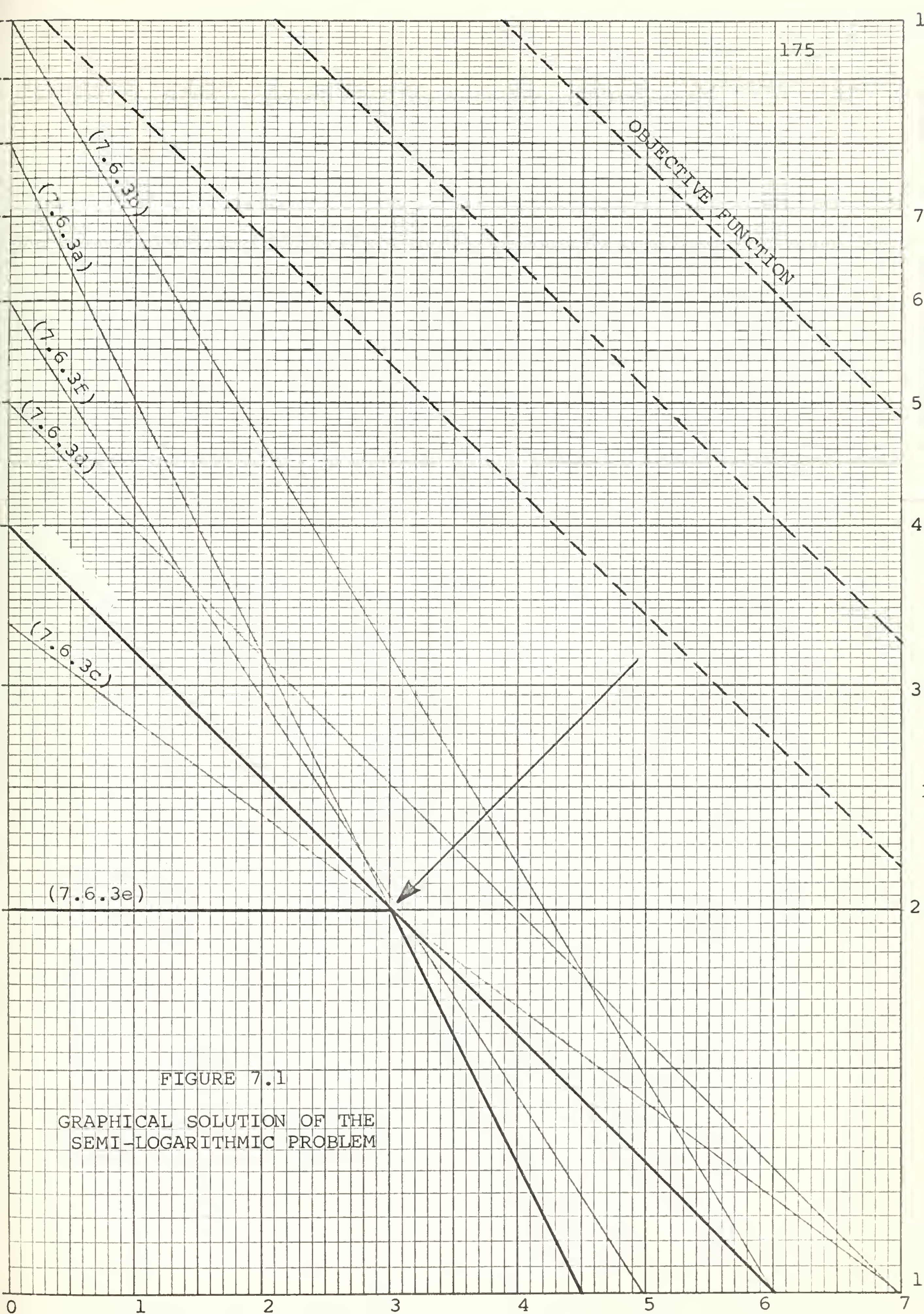


FIGURE 7.1

GRAPHICAL SOLUTION OF THE SEMI-LOGARITHMIC PROBLEM

to this problem to be:

$X_1 = 0.68945424$; $X_2 = 3.00799060$; FUNCTIONAL = 1.38444540.

Recalling that: $z = ye^{0.231x}$

$$X_1 = \ln y$$

$$X_2 = x$$

$$Z = \ln z$$

we can see, then:

$$\ln y = 0.68945424$$

or $$y = e^{0.68945424} = 1.99^{+}$$

and $$x = 3.0^{+}$$

$$Z = (1.99^{+})(e^{[0.231][3.00^{+}]})$$

$$Z = (1.99^{+})(1.99^{+}) = 4.0^{-}$$

But since the functional is given as: 1.384444540 we take the anti-logrithm and find it to be 3.99^{+}.

Comparing these results with the results achieved by the graphical method we find that they correspond as closely as can be distinguished. The conclusion, therefore, in this particular instance, is that the method employed has been successful.

It should be noted that this method must be restricted to positive values of the objective function (Z).

<u>Hyperbolic (Logarithmic) Maximization Problem</u>

Assume the following abstract system:

Objective (Maximize): $z = yx^{0.699}$

$y \leqq 10x^{-1.760}$ . . . . (7.7.1a)

$y \leqq 6x^{-1.113}$ . . . . (7.7.1b)

$y \leqq 7x^{-1.0}$ . . . . (7.7.1c)

$y \leqq 4x^{-0.602}$ . . . . (7.7.1d)

$y \leqq 2.5$ . . . . (7.7.1e)

$y \leqq 8x^{-1.474}$ . . . . (7.7.1f)

$y \geqq 0.0$

$x \geqq 0.0$

$z > 0.0$

By the use of logarithms the above system may be transformed into the following linear system:

Objective (Maximize): $Z = \ln z = \ln y + 0.699 \ln x$

$\ln y + 1.760 \ln x \leqq \ln 10$ . . . . (7.7.2a)

$\ln y + 1.113 \ln x \leqq \ln 6$ . . . . (7.7.2b)

$\ln y + \ln x \leqq \ln 7$ . . . . (7.7.2c)

$\ln y + 0.602 \ln x \leqq \ln 4$ . . . . (7.7.2d)

$\ln y \leqq \ln 2.5$ . . . . (7.7.2e)

$\ln y + 1.474 \ln x \leqq \ln 8$ . . . . (7.7.2f)

By making the following substitution:

$$\ln y = X_1$$

$$\ln x = X_2$$

the linear system, as plotted on logarithmic coordinate graph paper in figure 7.2, is as follows:

Objective (Maximize): $X_1 + 0.699X_2 = Z$

$$X_1 + 1.760X_2 \leqq 2.30259 \quad \text{(7.7.3a)}$$

$$X_1 + 1.113X_2 \leqq 1.79176 \quad \text{(7.7.3b)}$$

$$X_1 + X_2 \leqq 1.94591 \quad \text{(7.7.3c)}$$

$$X_1 + 0.602X_2 \leqq 1.38629 \quad \text{(7.7.3d)}$$

$$X_1 \leqq 0.91629 \quad \text{(7.7.3e)}$$

$$X_1 + 1.474X_2 \leqq 2.07944 \quad \text{(7.7.3f)}$$

The inequations may be converted to equations suitable for submission to the simplex algorithm by the introduction of slack variables, as follows:

$$X_1 + 0.699X_2 = Z \text{ (Maximize) Objective)}$$

$$X_1 + 1.760X_2 + X_3 = 2.30259$$

$$X_1 + 1.113X_2 + X_4 = 1.79176$$

$$X_1 + X_2 + X_5 = 1.94591$$

$$X_1 + 0.602X_2 + X_6 = 1.38629$$

$$X_1 + X_7 = 0.91629$$

$$X_1 + 1.474X_2 + X_8 = 2.07944$$

The above data, in the proper computer program format, is shown in Appendix II together with the computer solution to the problem. The computer output indicates the solution to this problem is:

$X_1 = 0.90984998$; $X_2 = 0.79135392$; FUNCTIONAL = 1.46298210.

Since $X_1 = \ln y$, and $X_2 = \ln x$, we take the anti-logrithms and find that $y = 2.489^{+}$ and $x = 2.209^{-}$ and the functional = 4.315+.

In this example, as in the previous one, the results obtained by the graphical method (figure 7.3) compare favorably, insofar as distinguishable, with the results obtained from the computer output. The conclusions to be drawn, therefore, would seem to be that, in this particular instance anyway, the method employed has been successful.

It should be noted that this method, as in the previous one, must be restricted to positive values of the objective function (Z).

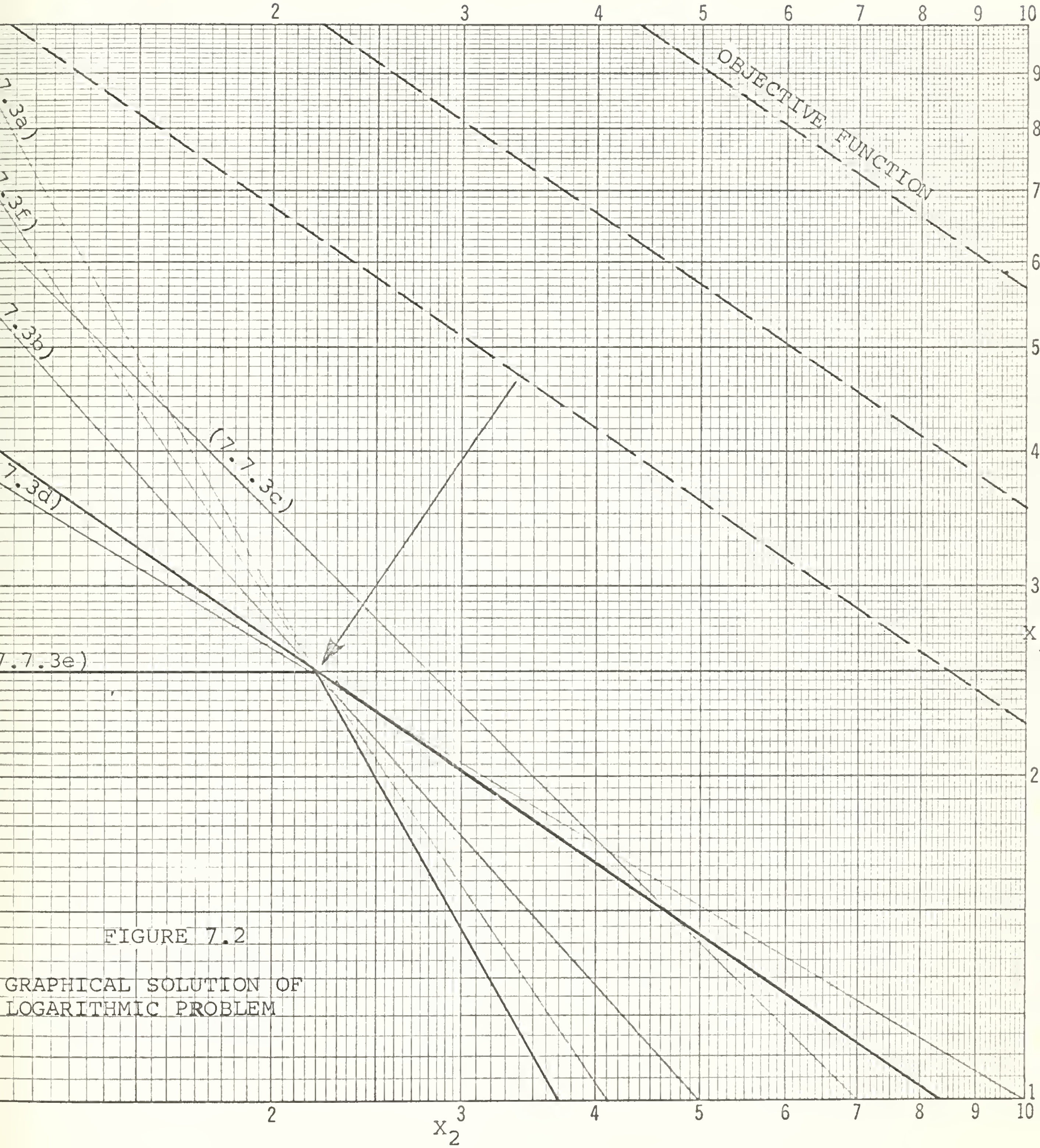


FIGURE 7.2

GRAPHICAL SOLUTION OF
LOGARITHMIC PROBLEM

CHAPTER VIII

SUMMARY AND CONCLUSIONS

An attempt has been made in this thesis to demonstrate the basic meaning implied in linear programming, and some of the ways this mathematical approach to solving resource management problems may be applied in private business as well as in military supply endeavors.

An emphasis has tried to be placed on the very important relationship existing between a mathematical description (or mathematical model) and the essence of what it describes, beyond any literal reality.

It has been shown, albeit with relatively simple examples, that linear programming most oftentimes deals with rather great quantities of arithmetic in solving problems. Problems that really need to be solved by linear programming, generally require the use of electronic digital computation equipment; otherwise, by the time some large problems could be solved, by hand computation, the problems would have become obsolete and a solution, even if accomplished, would be meaningless.

One should not have received the impression that linear programming is a panacea; it is not! Linear programming, like most everything else in life, has its disadvantages and faults.

Perhaps the greatest handicap (if one really could classify it as a handicap), to a potential user of linear programming, is that, almost invariably, a digital computer would be required to solve the problems which would actually be worth solving. In, and of itself this is not necessarily bad; it isn't bad, that is, if one always has a digital computer handy and the funds available with which to operate it. It is not unheard of that when some very large problems have been solved by linear programming, by means of a digital computer, the optimum solution indicated how savings could be made which were so small that they were insufficient to pay for the time and effort that had been expended in solving the problem, in the first place. It is not the intention, here, to downgrade linear programming; on the contrary, it should be understood that linear programming can be, and is, a very powerful and effective weapon against waste and inefficiency, but, like all new "wonder drugs", it must be used with discretion.

Those who are looking for an automatic decision making device will be disappointed with linear programming. To the best knowledge of this author there has not yet been developed a "device" or "technique" which can replace judgement and experience, or a "device" or technique" which will evaluate ethical or esthetic values and choices; at least, linear programming has not resolved these human dilemmas.

What linear programming can do, and do very well, is to solve certain types of mathematical models; and if the model has been properly constructed, and if proper data have been used (or, the limitations of the data are known), then linear programming can give an answer which will be the best (or the optimum) one under the particular set of circumstances.

Finally, if this thesis has helped to point out a direction in which one may go in looking for the best way to manage some resource, or many resources, (especially resources coming within the purview of the U. S. Navy Supply Corps) then, the predominate purpose of this thesis has been accomplished.

BIBLIOGRAPHY

BIBLIOGRAPHY

Books

Boulding, Kenneth E., W. Allen Spivey, et. al. Linear Programming and the Theory of the Firm. New York: The Macmillan Company, 1960.

Charnes, A. and W. W. Cooper. Management Models and Industrial Applications of Linear Programming. 2 vols. John Wiley and Sons, Inc., 1961.

Churchman, C. West, Russell L. Ackoff, E. Leonard Arnoff, et. al. Introduction to Operations Research. New York John Wiley and Sons, Inc., 1957.

Dorfman, Robert, Paul A. Samuelson and Robert M. Solow. Linear Programming and Economic Analysis. New York: McGraw-Hill Book Co., Inc., 1958.

Ferguson, Robert O. and Lauren F. Sargent. Linear Programming: Fundamentals and Applications. New York: McGraw-Hill Book Co., Inc., 1958.

Ficken, F. A. The Simplex Method of Linear Programming. New York: Holt, Rinehart and Winston, Inc., 1961.

Gale, David. The Theory of Linear Economic Models. New York: McGraw-Hill Book Co., Inc., 1960.

Garvin, Walter W. Introduction to Linear Programming. New York: McGraw-Hill Book Co., Inc., 1960.

Gass, Saul I. Linear Programming Methods and Applications. New York: McGraw-Hill Book Co., Inc., 1958.

Greenwald, Dakota Ulrich. Linear Programming: An Explanation of the Simplex Algorithm. New York: The Ronald Press Co., 1957.

Hadley, G. Linear Programming. Reading, Mass: Addison-Wesley Publishing Co., Inc., 1962.

Riley, Vera and Saul I. Gass. Linear Programming and Associated Techniques. Chevey Chase, Md.: Operations Research Office, The Johns Hopkins University, 1958.

Saaty, Thomas L. Mathematical Methods of Operations Research. New York: McGraw-Hill Book Co., Inc., 1959.

Vajda, S. An Introduction to Linear Programming and the Theory of Games. London: Methuen and Company, Ltd., 1960.

Vajda, S. Readings in Linear Programming. New York: John Wiley and Sons, Inc., 1958.

Periodicals

Acrivos, A. "Linear Programming. How Does it Work?" Chemical Engineering, 63:8:215-216, August, 1956.

Cooper, William W. and Abraham Charnes. "Linear Programming," Scientific American, 191:2:21-23, August, 1954.

Henderson, Alexander and Robert Schlaifer. "Mathematical Programming: Better Information for Better Decision Making," Harvard Business Review, (May-June, 1954), pp. 73-100.

Hetrick, James C. "Mathematical Models in Capital Budgeting," Harvard Business Review, (January-February, 1961), reprinted in HBR Statistical Decision Series, part II, pp. 55-70.

Payne, Bruce. "Steps in Long-Range Planning," Harvard Business Review, (March-April, 1957), pp. 95-106.

Rapoport, Leo A. and William P. Drews. "Mathematical Approach to Long-Range Planning," Harvard Business Review, (May-June, 1962), pp. 75-87.

Rosander, A. C. "The Use of Linear Programming to Improve the Quality of Decisions," Industrial Quality Control, 12:9:11-16, March, 1956.

Wrapp, Edward H. "Organization for Long-Range Planning," Harvard Business Review, (January-February, 1957), pp. 37-47.

APPENDIXES

APPENDIX I

COMPUTER PROGRAM TO DETERMINE APPROXIMATE OPTIMUM LOAD WEIGHT WITH MAXIMUM CARGO VALUE

Purpose

The purpose of this program is to compute the approximate optimum load weight (equal to or less than a specified maximum load weight limit) in order to obtain a maximum total value of cargo for the loading problem as explained in Chapter VII.

Language

Fortran II (IBM 1620 Computer).

Symbolic Dictionary

Variable	S/A*	I/O**	Description
RL	S	I&O	Maximum allowable load (or weight) limit of vehicle to be loaded.
N	S	I	Total number of items (or packages) to be considered for loading.
W	A	I	Weight of a package to be considered for loading.
V	A	I	Value of a package to be considered for loading.
X	A	--	An index. Computed internally as: $X_i = v_i/w_i$

---

*S - Single variable; A - Array of variables.

**I - Input; O - Output.

WI	A	0	Weight of item to be loaded.
VI	A	0	Value of item to be loaded.
XI	A	0	Index of item to be loaded except when WI = 0.0 and VI = 0.0 (see below).
WSUM	S	0	Total weight of cargo to be loaded.
VSUM	S	0	Total value of cargo to be loaded.

Program Routine

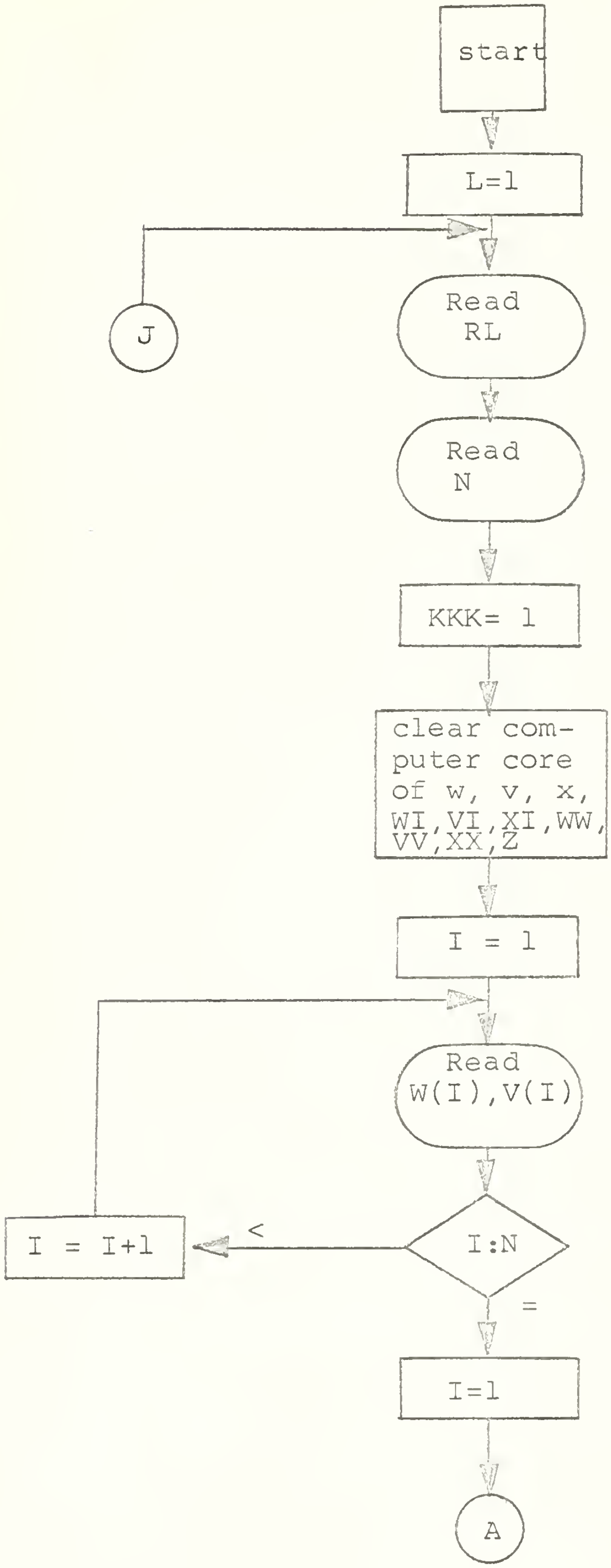
This program utilizes the data points (representing weights and values of packages, or items, to be loaded into a vehicle having a maximum cargo weight limit) to compute an index (X). By ordering the data in several ways and performing several checking procedures, a final approximate loading schedule is computed and is given as the output along with the total weight and value of all the packages to be included as cargo, and the maximum allowable weight (objective) of the vehicle. In some instances the loading schedule may contain weights and values of zero (0.0), but indicate an index number; these will be packages which were in a first feasible solution, based simply on the index criteria, and later replaced by a package through subsequent checking procedures. Only packages having weights and values greater than zero are to be considered in the final loading schedule.

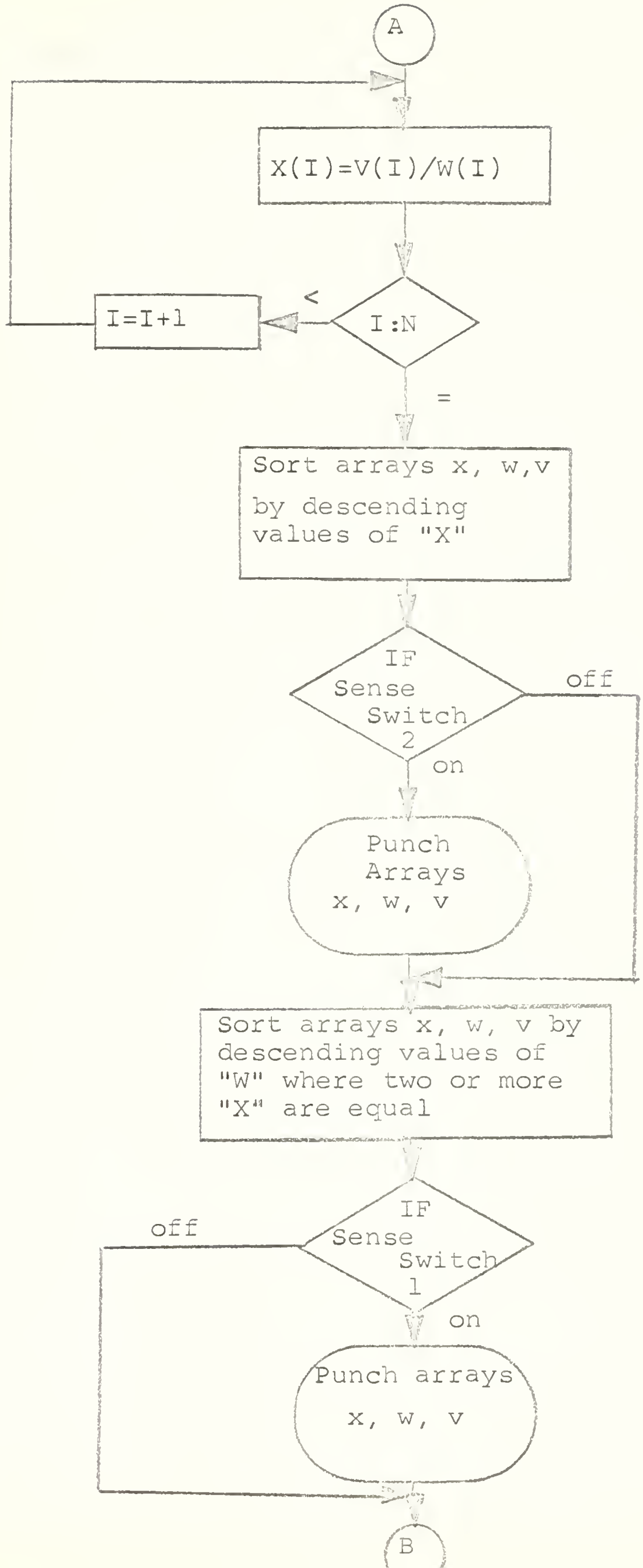
<u>Sense Switch Settings</u>

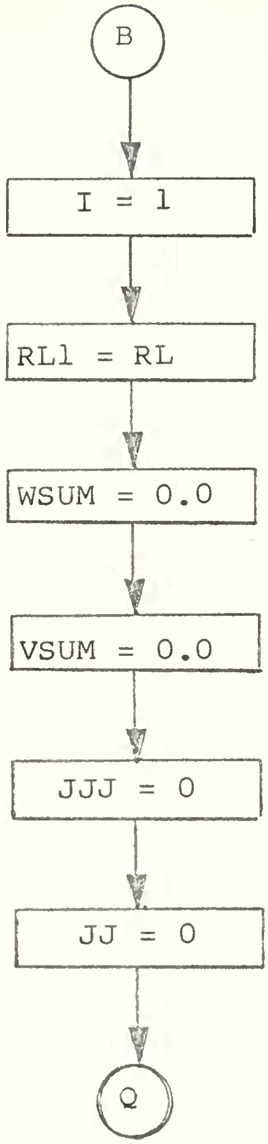
Sense Switch 1: When placed in the "on" position a listing of weights (w), values (v) and indexes (X) will be punched in descending order of index (X) and with descending order of weights (w) where two or more index numbers are the same.

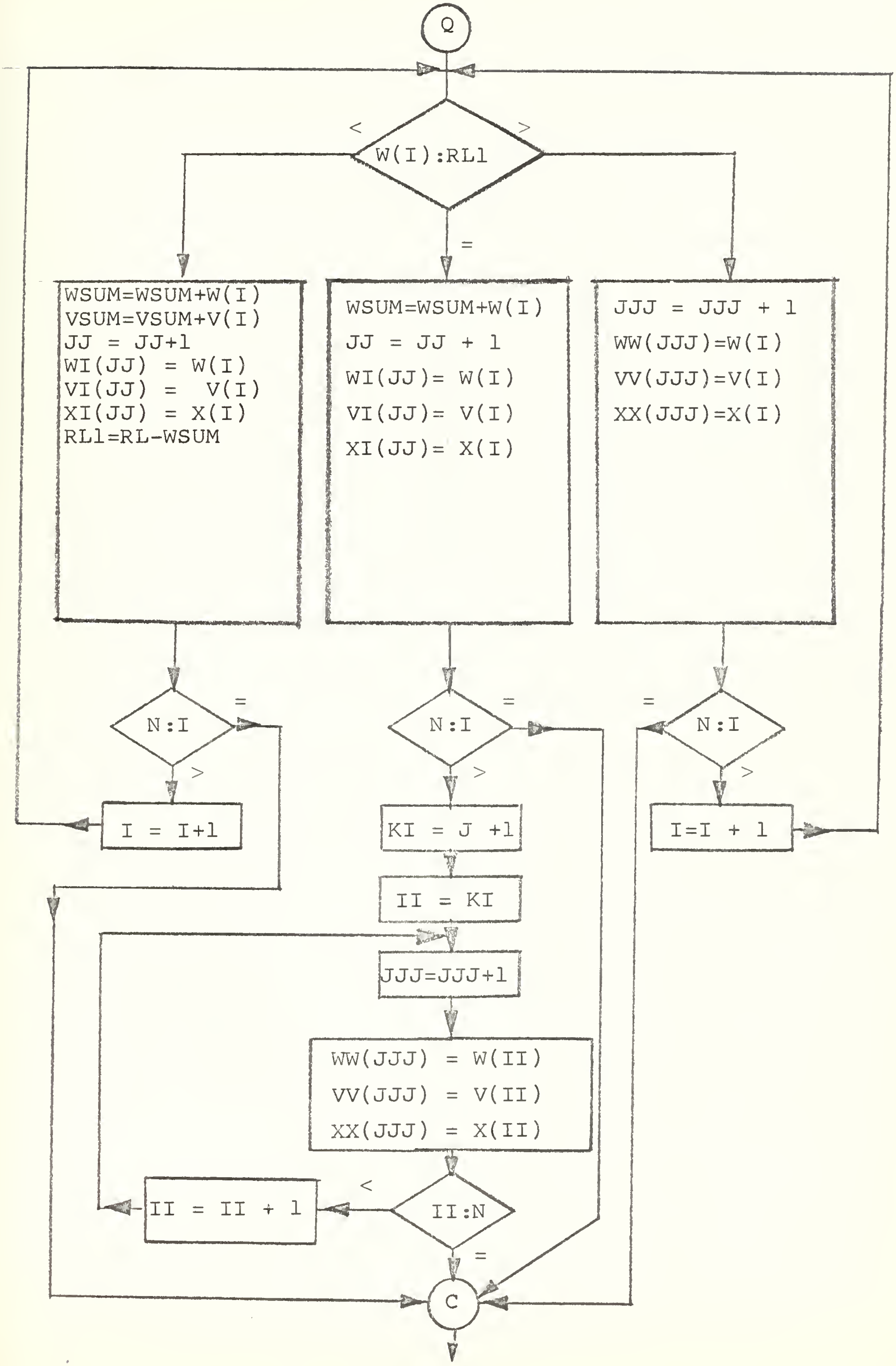
Sense Switch 2: When placed in the "on" position a listing of weights (w), values (v) and indexes (X) will be punched in descending order of index (X).

FLOW DIAGRAM FOR COMPUTER PROGRAM TO DETERMINE APPROXIMATE OPTIMUM LOAD WEIGHT WITH MAXIMUM CARGO VALUE

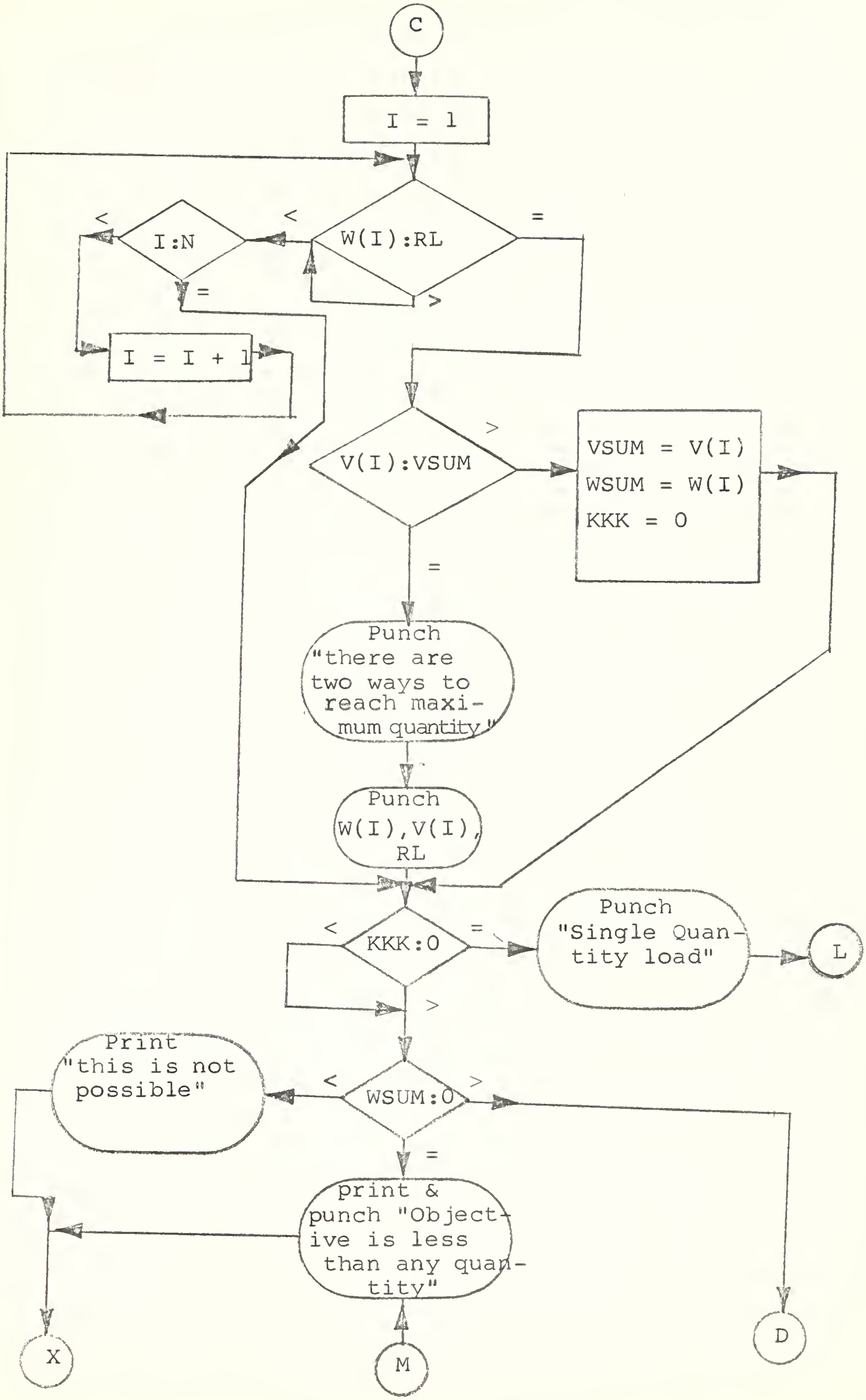


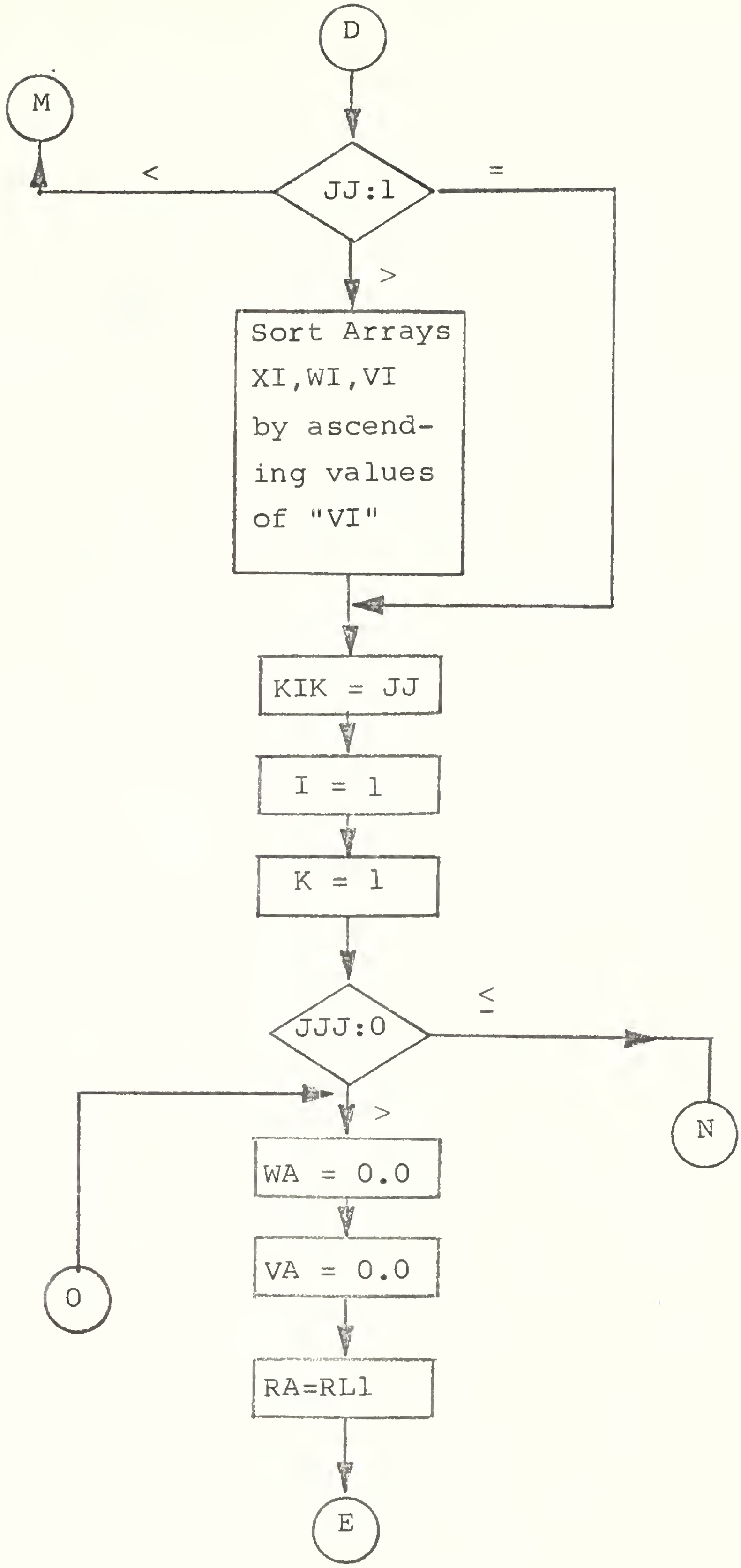

A
X(I)=V(I)/W(I)
I:N
<
I=I+1
=
Sort arrays x, w,v by descending values of "X"
IF Sense Switch 2
off
on
Punch Arrays x, w, v
Sort arrays x, w, v by descending values of "W" where two or more "X" are equal
IF Sense Switch 1
off
on
Punch arrays x, w, v
B

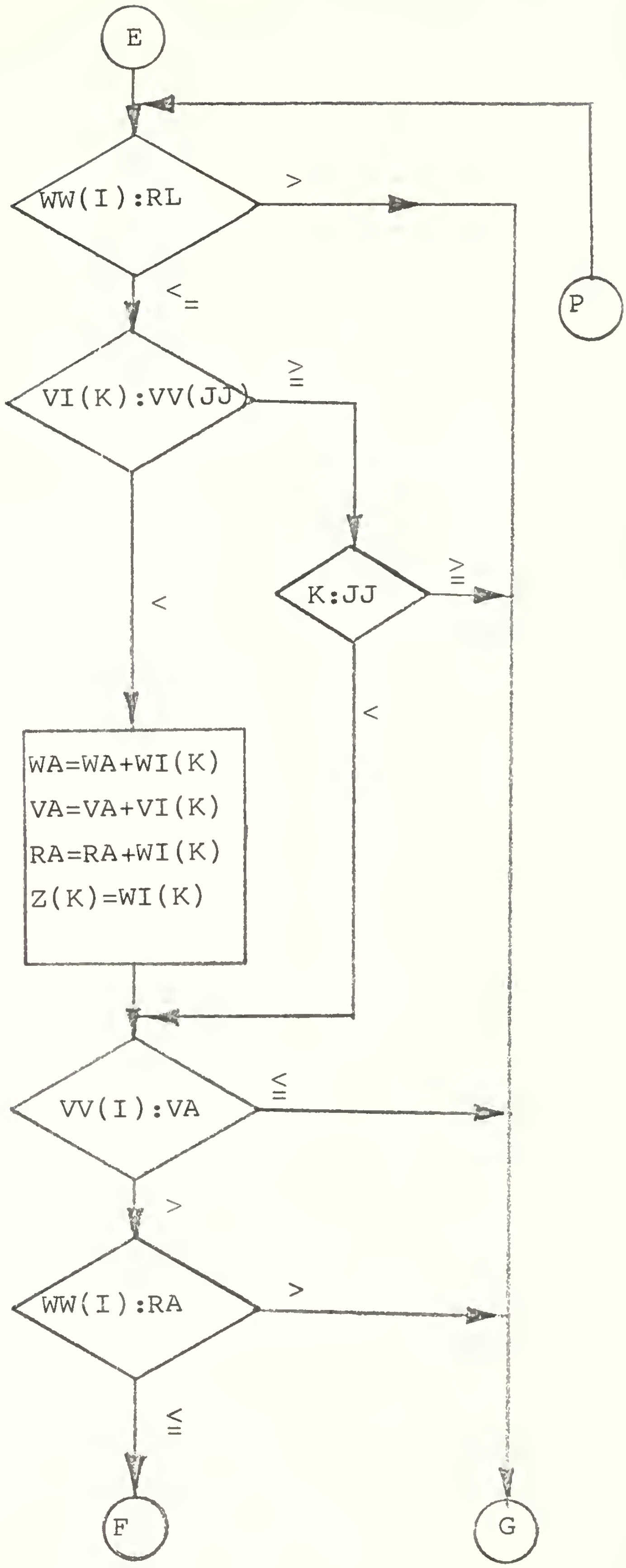

B
I = 1
RL1 = RL
WSUM = 0.0
VSUM = 0.0
JJJ = 0
JJ = 0
Q

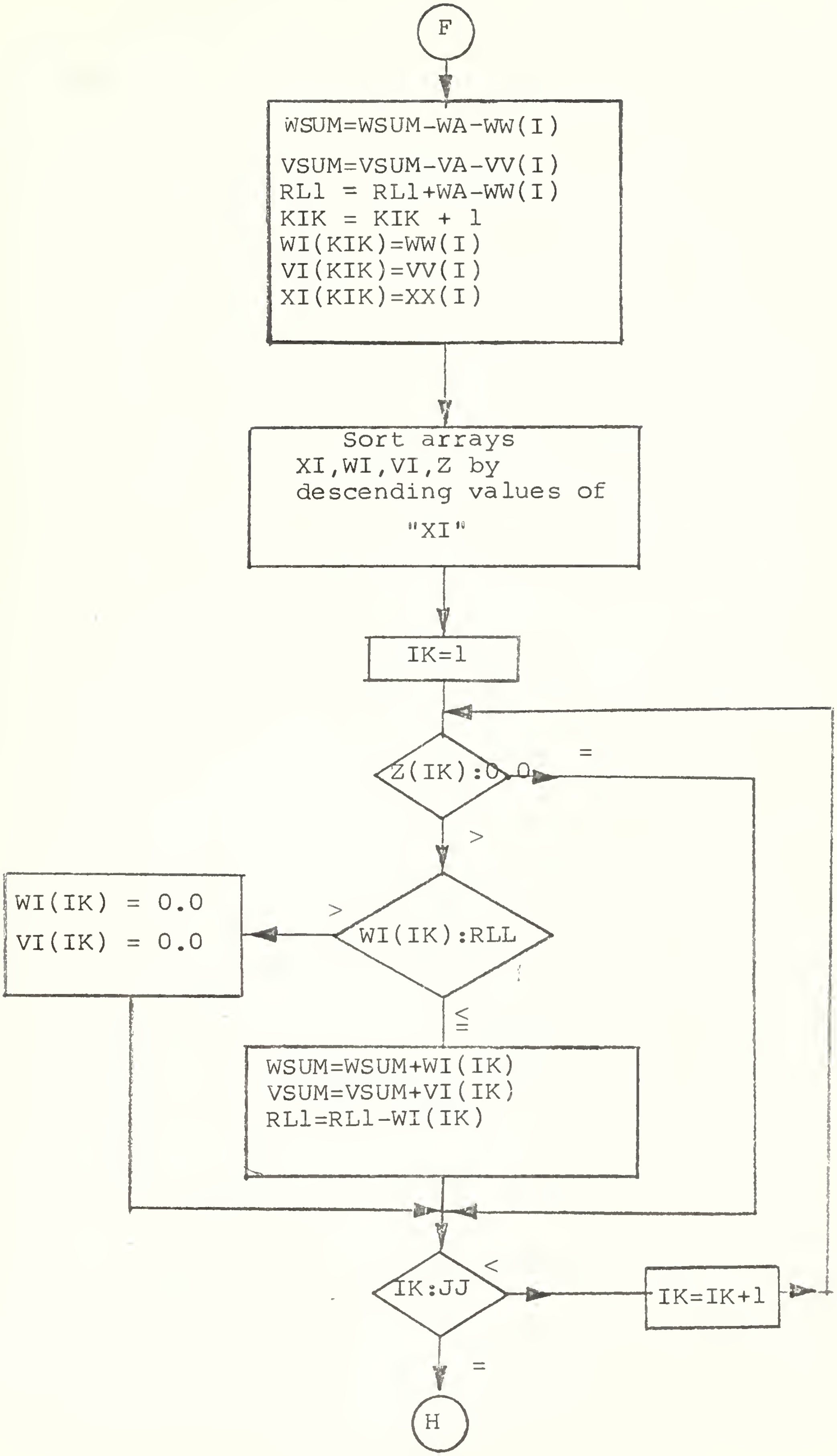


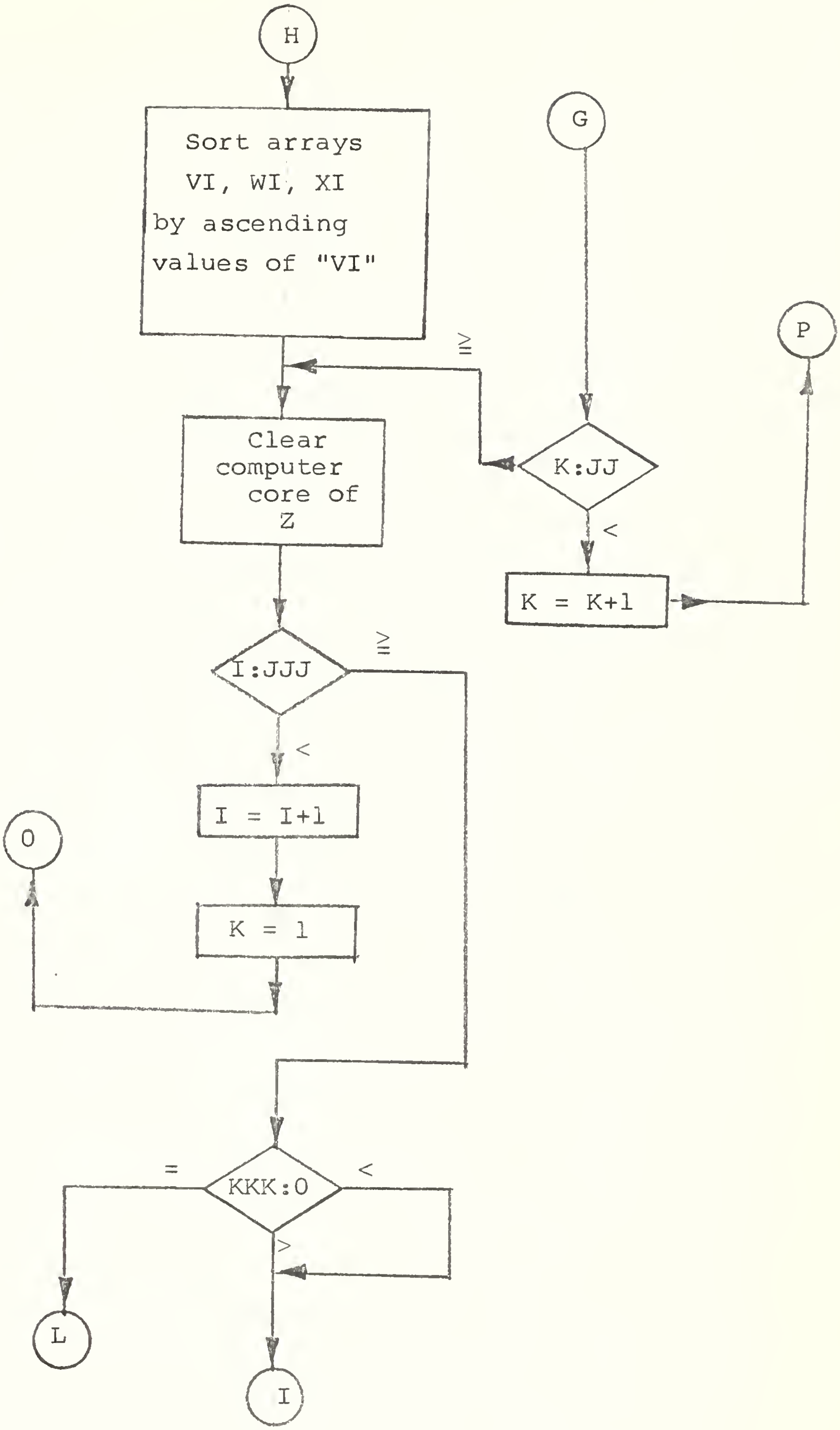
Q
W(I):RL1
<
>
=
WSUM=WSUM+W(I)
VSUM=VSUM+V(I)
JJ = JJ+1
WI(JJ) = W(I)
VI(JJ) = V(I)
XI(JJ) = X(I)
RL1=RL-WSUM
WSUM=WSUM+W(I)
JJ = JJ + 1
WI(JJ)= W(I)
VI(JJ)= V(I)
XI(JJ)= X(I)
JJJ = JJJ + 1
WW(JJJ)=W(I)
VV(JJJ)=V(I)
XX(JJJ)=X(I)
N:I
=
>
I = I+1
N:I
=
>
KI = J +1
II = KI
JJJ=JJJ+1
WW(JJJ) = W(II)
VV(JJJ) = V(II)
XX(JJJ) = X(II)
II:N
<
=
II = II + 1
N:I
=
>
I=I + 1
C

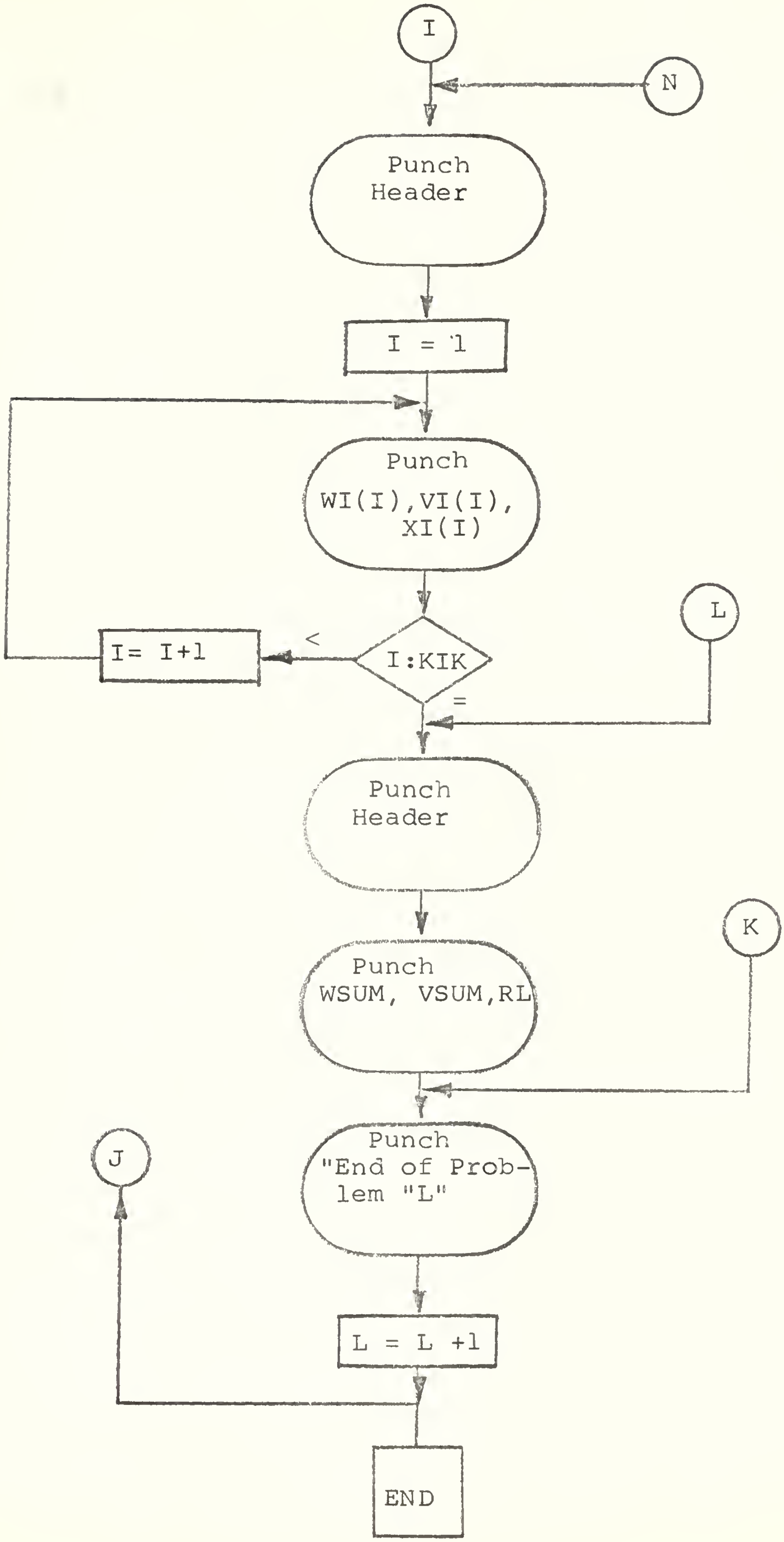

C
I = 1
W(I):RL
<
=
>
I:N
<
=
I = I + 1
V(I):VSUM
>
=
VSUM = V(I)
WSUM = W(I)
KKK = 0
Punch "there are two ways to reach maximum quantity"
Punch W(I),V(I), RL
KKK:0
<
=
>
Punch "Single Quantity load"
L
WSUM:0
<
>
=
Print "this is not possible"
print & punch "Objective is less than any quantity"
X
M
D


D
M
JJ:1
<
=
>
Sort Arrays XI,WI,VI by ascending values of "VI"
KIK = JJ
I = 1
K = 1
JJJ:0
≤
>
N
WA = 0.0
VA = 0.0
O
RA=RL1
E


E
WW(I):RL
>
P
<=
VI(K):VV(JJ)
≧
<
K:JJ
≧
<
WA=WA+WI(K)
VA=VA+VI(K)
RA=RA+WI(K)
Z(K)=WI(K)
VV(I):VA
≦
>
WW(I):RA
>
≦
F
G


F
WSUM=WSUM-WA-WW(I)
VSUM=VSUM-VA-VV(I)
RL1 = RL1+WA-WW(I)
KIK = KIK + 1
WI(KIK)=WW(I)
VI(KIK)=VV(I)
XI(KIK)=XX(I)
Sort arrays XI,WI,VI,Z by descending values of "XI"
IK=1
Z(IK):0.0
=
>
WI(IK):RLL
>
WI(IK) = 0.0
VI(IK) = 0.0
≤
WSUM=WSUM+WI(IK)
VSUM=VSUM+VI(IK)
RL1=RL1-WI(IK)
IK:JJ
<
IK=IK+1
=
H


H
Sort arrays
VI, WI, XI
by ascending
values of "VI"
G
P
≧
K:JJ
<
K = K+1
Clear
computer
core of
Z
I:JJJ
≧
<
I = I+1
O
K = 1
KKK:0
=
<
>
L
I


I
N
Punch
Header
I = 1
Punch
WI(I),VI(I),
XI(I)
L
I= I+1
<
I:KIK
=
Punch
Header
K
Punch
WSUM, VSUM,RL
J
Punch
"End of Prob-
lem "L"
L = L +1
END

```
C         PROGRAM TO DETERMINE APPROXIMATE OPTIMUM LOAD WEIGHT
C         WITH MAXIMUM CARGO VALUE
      DIMENSION W(50), V(50), X(50), WI(50), VI(50), XI(50)
      DIMENSION WW(50), VV(50), XX(50), Z(50)
      L = 1
   26 READ 10, RL
      READ 1, N
      KKK = 1
    1 FORMAT (I3)
    2 FORMAT (2F20.8)
   10 FORMAT (F20.8)
   99 FORMAT (1H //)
      PUNCH 99
      PUNCH 100, L
  100 FORMAT (21H BEGINNING OF PROBLEM,1X,I2)
      DO 70 I = 1, 50
      W(I) = 0.0
      V(I) = 0.0
      X(I) = 0.0
      WI(I) = 0.0
      VI(I) = 0.0
      XI(I) = 0.0
      WW(I) = 0.0
      VV(I) = 0.0
      XX(I) = 0.0
      Z(I) = 0.0
   70 CONTINUE
      READ 2, (W(I), V(I), I = 1,N)
      DO 3 I = 1,N
      X(I) = (V(I))/(W(I))
    3 CONTINUE
      K = N - 1
      DO 4 I = 1, K
      II = I + 1
      DO 4 J = II , N
      IF (X(I) - X(J)) 5, 5, 4
    5 TEMP = X(I)
      X(I) = X(J)
      X(J) = TEMP
      TEMP = V(I)
      V(I) = V(J)
      V(J) = TEMP
      TEMP = W(I)
      W(I) = W(J)
      W(J) = TEMP
    4 CONTINUE
      IF (SENSE SWITCH 2) 60, 61
   60 PUNCH 83
   83 FORMAT (12X, 31H FIRST SORT BY DESCENDING INDEX)
      PUNCH 35
      DO 61 I = 1,N
      PUNCH 14, X(I), W(I), V(I)
   61 CONTINUE
      PUNCH 99
```

```
      K = N - 1
      DO 7 I = 1.K
      II = I + 1
      DO 7 J = II.N
      IF (X(I) - X(J)) 7. 8. 7
    8 IF (W(I) - W(J)) 9.7.7
    9 TEMP = W(I)
      W(I) = W(J)
      W(J) = TEMP
      TEMP = V(I)
      V(I) = V(J)
      V(J) = TEMP
    7 CONTINUE
      IF(SENSE SWITCH 1) 11. 13
   11 PUNCH 84
   84 FORMAT (3X. 44H SECOND SORT BY DESCENDING QUANTITY HAVI.20HNG IDEN
  841TICAL INDEXES)
      PUNCH 35
   35 FORMAT (8X. 6H INDEX. 15X. 9H QUANTITY. 19X. 6H VALUE)
      DO 12 I = 1.N
   13 PUNCH 14. X(I). W(I). V(I)
   14 FORMAT (F20.8. 4X. F20.8. 4X. F20.8)
   12 CONTINUE
      PUNCH 99
      I = 1
      RL1 = RL
      WSUM = 0.0
      VSUM = 0.0
      JJJ = 0
      JJ = 0
   18 IF(W(I) - RL1) 15. 21. 20
   15 WSUM = WSUM + W(I)
      VSUM = VSUM + V(I)
      JJ = JJ + 1
      WI(JJ) = W(I)
      VI(JJ) = V(I)
      XI(JJ) = X(I)
      RL1 = RL - WSUM
   23 IF(N-I) 16. 16. 19
   19 I = I + 1
      GO TO 18
   20 JJJ = JJJ + 1
      WW(JJJ) = W(I)
      VV(JJJ) = V(I)
      XX(JJJ) = X(I)
      IF(N-I) 16. 25. 17
   17 I = I + 1
      GO TO 18
   21 WSUM = WSUM + W(I)
      VSUM = VSUM + V(I)
      JJ = JJ + 1
      WI(JJ) - W(I)
      VI(JJ) = V(I)
      XI(JJ) = X(I)
```

```
 16 IF(N - I) 205, 205, 204
204 KI = I + 1
    DO 200 II = KI, N
    JJJ = JJJ + 1
    WW(JJJ) = W(II)
    VV(JJJ) = V(II)
    XX(JJJ) = X(II)
200 CONTINUE
205 DO 40 I = 1, N
    IF (W(I) - RL) 40, 42, 40
 42 IF(V(I) - VSUM) 40, 44, 45
 44 PUNCH 46
 46 FORMAT (45H THERE ARE TWO WAYS TO REACH MAXIMUM QUANTITY)
    PUNCH 99
    PUNCH 25
    PUNCH 22, W(I), V(I), RL
    PUNCH 99
    GO TO 40
 45 VSUM = V(I)
    WSUM = W(I)
    KKK = 0
    PUNCH 92
 92 FORMAT (21H SINGLE QUANTITY LOAD)
 40 CONTINUE
    IF(KKK) 241, 55, 241
241 IF(WSUM) 75, 76, 77
 75 PRINT 78
 78 FORMAT (21H THIS IS NOT POSSIBLE)
    GO TO 88
 76 PRINT 79
    PUNCH 79
 79 FORMAT (36H OBJECTIVE IS LESS THAN ANY QUANTITY)
    GO TO 88
 77 CONTINUE
    IF(JJ -1) 75, 229, 228
228 IJI = JJ - 1
    DO 212 I = 1, IJI
    KPI = I + 1
    DO 212 J = KPI, JJ
    IF(VI(I) - VI(J)) 212, 212, 213
213 TEMP = XI(I)
    XI(I) = XI(J)
    XI(J) = TEMP
    TEMP = WI(I)
    WI(I) = WI(J)
    WI(J) = TEMP
    TEMP = VI(I)
    VI(I) = VI(J)
    VI(J) = TEMP
212 CONTINUE
229 KIK = JJ
    I = 1
    K = 1
    IF(JJJ) 90, 90, 230
```

```
222 [illegible] = [illegible]
    [illegible] = [illegible]
    [illegible] = PLI
    IF([illegible]) [illegible]
    IF([illegible]) [illegible]
    IF([illegible]) [illegible]
    [illegible] = [illegible]
    [illegible] = [illegible]
    [illegible] = [illegible]
    [illegible] = [illegible]
225 IF([illegible]) [illegible]
226 IF([illegible]) [illegible]
    [illegible] = [illegible]
    [illegible] = [illegible]
    [illegible] = [illegible]
    [illegible] = [illegible]
    [illegible] = [illegible]
    [illegible] = [illegible]
    [illegible] = [illegible]
    [illegible] = [illegible]
    DO [illegible]
    [illegible] = I + 1
    DO [illegible]
    IF([illegible]) [illegible]
    TEMP = [illegible]
    [illegible] = [illegible]
    [illegible] = TEMP
    TEMP = [illegible]
    [illegible] = [illegible]
    [illegible] = TEMP
    TEMP = [illegible]
    [illegible] = [illegible]
    [illegible] = TEMP
    TEMP = [illegible]
    [illegible] = [illegible]
    [illegible] = TEMP
    CONTINUE
    [illegible]
    [illegible]
    IF([illegible]) [illegible]
    [illegible]
    [illegible]
    [illegible]
    [illegible]
    [illegible] = [illegible]
    [illegible] = [illegible]
    CONTINUE
    [illegible]
    DO [illegible]
    [illegible]
    DO [illegible]
    IF([illegible]) [illegible]
    TEMP = [illegible]
    [illegible] = [illegible]
```

```
232 WA = 0.0
    VA = 0.0
    RA = RL1
260 IF(WW(I) - RL) 233, 233, 218
233 IF(VI(K) - VV(I)) 223, 224, 224
224 IF(K - JJ) 218, 225, 225
223 WA = WA + WI(K)
    VA = VA + VI(K)
    RA = RA + WI(K)
    Z(K) = WI(K)
225 IF(VV(I) - VA) 218, 218, 226
226 IF(WW(I) - RA) 217, 217, 218
217 WSUM = WSUM - WA + WW(I)
    VSUM = VSUM - VA + VV(I)
    RL1 = RL1 + WA - WW(I)
    KIK = KIK + 1
    WI(KIK) = WW(I)
    VI(KIK) = VV(I)
    XI(KIK) = XX(I)
    IJI = JJ - 1
    DO 252 I = 1, IJI
    KPI = I + 1
    DO 252 J = KPI, JJ
    IF(XI(I) - XI(J)) 253, 253, 252
253 TEMP = XI(I)
    XI(I) = XI(J)
    XI(J) = TEMP
    TEMP = WI(I)
    WI(I) = WI(J)
    WI(J) = TEMP
    TEMP = VI(I)
    VI(I) = VI(J)
    VI(J) = TEMP
    TEMP = Z(I)
    Z(I) = Z(J)
    Z(J) = TEMP
252 CONTINUE
    DO 221 IK = 1, JJ
    IF(Z(IK)) 221, 221, 249
249 IF(WI(IK) - RL1) 248, 248, 222
248 WSUM = WSUM + WI(IK)
    VSUM = VSUM + VI(IK)
    RL1 = RL1 - WI(IK)
    GO TO 221
222 WI(IK) = 0.0
    VI(IK) = 0.0
221 CONTINUE
    IJI = JJ - 1
    DO 256 I = 1, IJI
    KPI = I + 1
    DO 256 J = KPI, JJ
    IF(VI(I) - VI(J)) 256, 256, 258
258 TEMP = XI(I)
    XI(I) = XI(J)
```

```
      XI(J) = TEMP
      TEMP = VI(I)
      VI(I) = VI(J)
      VI(J) = TEMP
      TEMP = WI(I)
      WI(I) = WI(J)
      WI(J) = TEMP
  256 CONTINUE
      GO TO 219
  218 IF(K - JJ) 215, 216, 216
  219 CONTINUE
      DO 235 IIK = 1, JJ
      Z(IIK) = 0.0
  235 CONTINUE
  216 IF(I - JJJ) 236, 237, 237
  236 I = I + 1
      K = 1
      GO TO 232
  215 K = K + 1
      GO TO 260
  237 CONTINUE
      IF(KKK) 90, 55, 90
   90 PUNCH 300
  300 FORMAT(33X, 17H LOADING SCHEDULE)
      PUNCH 50
   50 FORMAT (7X, 9H QUANTITY, 18X, 6H VALUE, 18X, 6H INDEX)
      DO 55 I = 1, KIK
      PUNCH 22, WI(I), VI(I), XI(I)
   55 CONTINUE
      PUNCH 99
      PUNCH 25
   25 FORMAT(3X,13H QUANTITY [illegible] VALUE [illegible],18X,1[illegible]H OBJECTIVE)
      PUNCH 22, WSUM, VSUM, BL
   22 FORMAT (F20.8, 4X, F20.8, 4X, F20.8)
      PUNCH 99
   88 CONTINUE
      PUNCH 98, L
   98 FORMAT (15H [illegible],1X,I[illegible])
      L = L + 1
      GO TO 26
      END
```

COMPUTER PROGRAM TO DETERMINE APPROXIMATE OPTIMUM LOAD WEIGHT, AND LOAD VOLUME, WITH MAXIMUM CARGO VALUE

Purpose

The purpose of this program is to compute the approximate optimum load weight and load value (equal to or less than a specified maximum load weight limit and volume limit) in order to obtain a maximum total value of cargo for the loading problem with volume considerations as explained in chapter VII.

Language

Fortran II (IBM 1620 Computer).

Symbolic Dictionary

Variable	S/A*	I/O**	Description
RL	S	I&O	Maximum allowable weight limit of vehicle to be loaded.
CL	S	I&O	Maximum allowable volume limit of vehicle to be loaded.
N	S	I	Total number of items (or packages) to be considered for loading.
W	A	I	Weight of a package to be considered for loading.
V	A	I	Value of a package to be considered for loading.
C	A	I	Volume of a package to be considered for loading.

---

*S - Single variable; A - Array of variables

**I - Input; O - output

X	A	--	An Index. Computed internally as: $x_i = v_i/(w_i)(c_i)$
WI	A	0	Weight of package to be loaded.
VI	A	0	Value of package to be loaded.
CI	A	0	Volume of package to be loaded.
XI	A	0	Index of package to be loaded except when WI = 0.0, VI = 0.0 and CI = 0.0 (see below).
WSUM	S	0	Total weight of cargo to be loaded.
VSUM	S	0	Total value of cargo to be loaded.
CSUM	S	0	Total volume of cargo to be loaded.

Program Routing

This program utilizes the data points (representing weights, values, and volumes of packages, or items, to be loaded into a vehicle having a maximum cargo weight limit and a maximum cargo volume limit) to compute on index (X). By ordering the data in several ways and performing several checking procedures, a final approximate loading schedule is computed and is given as the output along with the total weight, total value and total volume of all the packages to be included as cargo, and the maximum allowable weight (objective) and maximum allowable volume (objective) of the vehicle. In some instances the loading schedule may contain weights, values, and volumes of zero (0.0), but indicate an index number; these will be packages

which were in a first feasible solution, based, simply, on the index criteria, and later replaced by a package through subsequent checking procedures. Only packages having weights and values greater than zero are to be considered in the final loading schedule.

Sense Switch Settings

Sense Switch 1: When placed in the "on" position a listing of weights (w), values (v), and volumes will be punched in descending order of index (X) and with descending order of weights (w) where two or more index numbers are the same.

```
C     PROGRAM TO DETERMINE APPROXIMATE OPTIMUM LOAD WEIGHT,
C     AND LOAD VOLUME WITH MAXIMUM CARGO VALUE
      DIMENSION W(50), V(50), X(50), WI(50), VI(50), XI(50)
      DIMENSION WW(50), VV(50), XX(50), Z(50), CC(50)
      DIMENSION C(50), CI(50)
      L = 1
   26 READ 10, RL, CL
      READ 1, N
      KKK = 1
    1 FORMAT (I3)
    2 FORMAT (3F15.4)
   10 FORMAT (2F15.4)
   99 FORMAT (1H //)
      PUNCH 99
      PUNCH 100, L
  100 FORMAT (21H BEGINNING OF PROBLEM,1X,I2)
      DO 70 I = 1, 50
      W(I) = 0.0
      V(I) = 0.0
      C(I) = 0.0
      X(I) = 0.0
      WI(I) = 0.0
      VI(I) = 0.0
      CI(I) = 0.0
      XI(I) = 0.0
      WW(I) = 0.0
      VV(I) = 0.0
      XX(I) = 0.0
      CC(I) = 0.0
      Z(I) = 0.0
   70 CONTINUE
      READ 2, (W(I), V(I), C(I), I = 1, N)
      DO 3 I = 1, N
      X(I) = V(I)/(W(I)*C(I))
    3 CONTINUE
      K = N - 1
      DO 4 I = 1, K
      II = I + 1
      DO 4 J = II, N
      IF (X(I) - X(J)) 5, 5, 4
    5 TEMP = X(I)
      X(I) = X(J)
      X(J) = TEMP
      TEMP = V(I)
      V(I) = V(J)
      V(J) = TEMP
      TEMP = W(I)
      W(I) = W(J)
      W(J) = TEMP
      TEMP = C(I)
      C(I) = C(J)
      C(J) = TEMP
    4 CONTINUE
      IF (SENSE SWITCH 1) 60, 61
```

```
60 PUNCH 83
83 FORMAT (12X, 31H FIRST SORT BY DESCENDING INDEX)
   PUNCH 35
   DO 61 I = 1, N
   PUNCH 14, X(I), W(I), C(I), V(I)
61 CONTINUE
   PUNCH 99
   K = N - 1
   DO 7 I = 1, K
   II = I + 1
   DO 7 J = II, N
   IF (X(I) - X(J)) 7, 8, 7
 8 IF(V(I) - V(J)) 9, 7, 7
 9 TEMP = W(I)
   W(I) = W(J)
   W(J) = TEMP
   TEMP = V(I)
   V(I) = V(J)
   V(J) = TEMP
   TEMP = C(I)
   C(I) = C(J)
   C(J) = TEMP
 7 CONTINUE
   IF (SENSE SWITCH 1) 11, 12
11 PUNCH 84
84 FORMAT(3X,40H SECOND SORT BY DESCENDING VALUES HAVING,18H IDENTICA
841L INDEXES)
   PUNCH 35
35 FORMAT(8X,6H INDEX,11X,7H WEIGHT,14X,5H CUBE,14X,6H VALUE)
   DO 12 I = 1, N
13 PUNCH 14, X(I),W(I),C(I),V(I)
14 FORMAT(1X,F15.4,4X,F15.4,4X,F15.4,4X,F15.4)
12 CONTINUE
   PUNCH 99
   I = 1
   RL1 = RL
   CL1 = CL
   WSUM = 0.0
   VSUM = 0.0
   CSUM = 0.0
   JJJ = 0
   JJ = 0
18 IF(W(I) - RL1) 115, 121, 19
115 IF(C(I) - CL1) 15, 21, 19
15 WSUM = WSUM + W(I)
   VSUM = VSUM + V(I)
   CSUM = CSUM + C(I)
   JJ = JJ + 1
   WI(JJ) = W(I)
   VI(JJ) = V(I)
   CI(JJ) = C(I)
   XI(JJ) = X(I)
   RL1 = RL - WSUM
   CL1 = CL - CSUM
```

```
      IF(RL1) 16, 16, 123
  123 IF(CL1) 16, 16, 239
  239 IF(N-I) 16, 16, 240
  240 I = I + 1
      GO TO 18
   19 JJJ = JJJ + 1
      WW(JJJ) = W(I)
      VV(JJJ) = V(I)
      XX(JJJ) = X(I)
      CC(JJJ) = C(I)
      IF(N-I) 16, 205, 23
   23 I = I + 1
      GO TO 18
  121 IF(C(I) - CL1) 21, 21, 19
   21 WSUM = WSUM + W(I)
      VSUM = VSUM + V(I)
      CSUM = CSUM + C(I)
      JJ = JJ + 1
      WI(JJ) = W(I)
      VI(JJ) = V(I)
      CI(JJ) = C(I)
      XI(JJ) = X(I)
   16 IF(N - I) 205, 205, 204
  204 KI = I + 1
      DO 200 II = KI, N
      JJJ = JJJ + 1
      WW(JJJ) = W(II)
      VV(JJJ) = V(II)
      XX(JJJ) = X(II)
      CC(JJJ) = C(II)
  200 CONTINUE
  205 DO 40 I = 1, N
      IF(W(I) - RL) 40, 142, 40
  142 IF(C(I) - CL) 42, 42, 40
   42 IF(V(I) - VSUM) 40, 44, 45
   44 PUNCH 46
   46 FORMAT (45H THERE ARE TWO WAYS TO REACH MAXIMUM QUANTITY)
      PUNCH 99
      PUNCH 25
      PUNCH 22, W(I), RL, C(I), CL
      PUNCH 145, V(I)
      PUNCH 99
      GO TO 40
   45 VSUM = V(I)
      WSUM = W(I)
      CSUM = C(I)
      KKK = 0
      PUNCH 92
   92 FORMAT (21H SINGLE QUANTITY LOAD)
   40 CONTINUE
      IF(KKK) 241, 55, 241
  241 IF(WSUM) 75, 76, 75
   75 PRINT 78
      PUNCH 78
```

```
 78 FORMAT (21H THIS IS NOT POSSIBLE)
    GO TO 88
 76 PUNCH 79
    PRINT 79
 79 FORMAT (36H OBJECTIVE IS LESS THAN ANY QUANTITY)
    GO TO 88
 77 CONTINUE
    IF(CSUM) 75, 76, 177
177 CONTINUE
    IF(JJ - 1) 76, 229, 228
228 IJI = JJ - 1
    DO 212 I = 1, IJI
    KPI = I + 1
    DO 212 J = KPI, JJ
    IF(VI(I) - VI(J)) 212, 212, 213
213 TEMP = XI(I)
    XI(I) = XI(J)
    XI(J) = TEMP
    TEMP = WI(I)
    WI(I) = WI(J)
    WI(J) = TEMP
    TEMP = VI(I)
    VI(I) = VI(J)
    VI(J) = TEMP
    TEMP = CI(I)
    CI(I) = CI(J)
    CI(J) = TEMP
212 CONTINUE
229 KIK = JJ
    I = 1
    K = 1
    IF(JJJ) 90, 91, 232
232 WA = 0.0
    VA = 0.0
    CA = 0.0
    RA = RL1
    CR = CL1
260 IF(WW(I) - RL) 270, 270, 218
270 IF(CC(I) - CL) 233, 233, 218
233 IF(VI(K) - VV(I)) 223, 224, 224
224 IF(K - JJ) 218, 225, 225
223 WA = WA + WI(K)
    VA = VA + VI(K)
    CA = CA + CI(K)
    RA = RA + WI(K)
    CR = CR + CI(K)
    Z(K) = WI(K)
225 IF(VV(I) - VA) 218, 218, 226
226 IF(WW(I) - RA) 217, 217, 218
217 IF(CC(I) - CR) 227, 227, 218
227 WSUM = WSUM - WA + WW(I)
    VSUM = VSUM - VA + VV(I)
    CSUM = CSUM - CA + CC(I)
    RL1 = RL1 + WA - WW(I)
```

```
      CL1 = CL1 + CA - CC(I)
      KIK = KIK + 1
      WI(KIK) = WW(I)
      VI(KIK) = VW(I)
      XI(KIK) = XX(I)
      CI(KIK) = CC(I)
      IJI = JJ - 1
      DO 252 I = 1, IJI
      KPI = I + 1
      DO 252 J = KPI, JJ
      IF(XI(I) - XI(J)) 253, 253, 252
  253 TEMP = XI(I)
      XI(I) = XI(J)
      XI(J) = TEMP
      TEMP = WI(I)
      WI(I) = WI(J)
      WI(J) = TEMP
      TEMP = VI(I)
      VI(I) = VI(J)
      VI(J) = TEMP
      TEMP = CI(I)
      CI(I) = CI(J)
      CI(J) = TEMP
      TEMP = Z(I)
      Z(I) = Z(J)
      Z(J) = TEMP
  252 CONTINUE
      DO 221 IK = 1, JJ
      IF(Z(IK)) 221, 221, 249
  249 IF(WI(IK) - RL1) 280, 280, 222
  280 IF(CI(IK) - CL1) 248, 248, 222
  248 WSUM = WSUM + WI(IK)
      VSUM = VSUM + VI(IK)
      RL1 = RL1 - WI(IK)
      CL1 = CL1 - CI(IK)
      GO TO 221
  222 WI(IK) = 0.0
      VI(IK) = 0.0
      CI(IK) = 0.0
  221 CONTINUE
      IJI = JJ - 1
      DO 256 I = 1, IJI
      KPI = I + 1
      DO 256 J = KPI, JJ
      IF(VI(I) - VI(J)) 256, 258, 258
  258 TEMP = XI(I)
      XI(I) = XI(J)
      XI(J) = TEMP
      TEMP = VI(I)
      VI(I) = VI(J)
      VI(J) = TEMP
      TEMP = WI(I)
      WI(I) = WI(J)
      WI(J) = TEMP
```

```
      TEMP = CI(I)
      CI(I) = CI(J)
      CI(J) = TEMP
  256 CONTINUE
      GO TO 219
  218 IF(K - JJ) 215, 219, 219
  219 CONTINUE
      DO 235 IIK = 1, JJ
      Z(IIK) = 0.0
  235 CONTINUE
  216 IF(I - JJJ) 236, 237, 237
  236 I = I + 1
      K = 1
      GO TO 232
  215 K = K + 1
      GO TO 260
  237 CONTINUE
      IF(KKK) 90, 55, 88
   90 PUNCH 150
  150 FORMAT (33X,17H LOADING SCHEDULE)
      PUNCH 50
   50 FORMAT(7X,7H WEIGHT,13X,5H CUBE,14X,6H VALUE,13X,6H INDEX)
      DO 55 I = 1, KIK
      PUNCH 22, WI(I),CI(I),VI(I),XI(I)
   55 CONTINUE
      PUNCH 99
      PUNCH 25
   25 FORMAT(15H MAXIMUM WEIGHT,3X,17H WEIGHT OBJECTIVE,4X,13H MAXIMUM C
   251UBE,5X,15H CUBE OBJECTIVE)
      PUNCH 22, WSUM,RL, CSUM, CL
   22 FORMAT(F15.4,4X,F15.4,4X,F15.4,4X,F15.4)
      PUNCH 145, VSUM
  145 FORMAT (25X,14H CARGO VALUE =, F15.4)
      PUNCH 99
   88 CONTINUE
      PUNCH 98, L
   98 FORMAT(15H END OF PROBLEM,1X,I2)
      L = L + 1
      GO TO 26
      END
```

COMPUTER PROGRAM TO GENERATE RANDOM DATA

Purpose

The purpose of this program is to generate simulated data for use in testing the approximation algorithm (i.e., the computer program to determine approximate optimum load weight with maximum cargo value). The data produced by this program is suitable for direct submission to either the above mentioned program or to the appropriate direct enumeration program, which follows:

Language

Fortran II. (IBM 1620 Computer).

Symbolic Dictionary

Variable	S/A*	I/O**	Description
NR	A	I	Table of random numbers (1040, nine digit, fixed point, numbers).
KK	S	I	Number of data points (or simulated packages) desired as output data.
L	S	I	Any random number between one (1) and 1039, indicating where the program is to begin in the random number table.

---

*S - Single variable; A - Array of variables.

**I - Input; O - Output.

NSETS	S	I	Number of sets of data desired.
AX	S	0	Simulated maximum allowable weight limit.
AW	A	0	Simulated package weights.
AV	A	0	Simulated package values

Program Routine

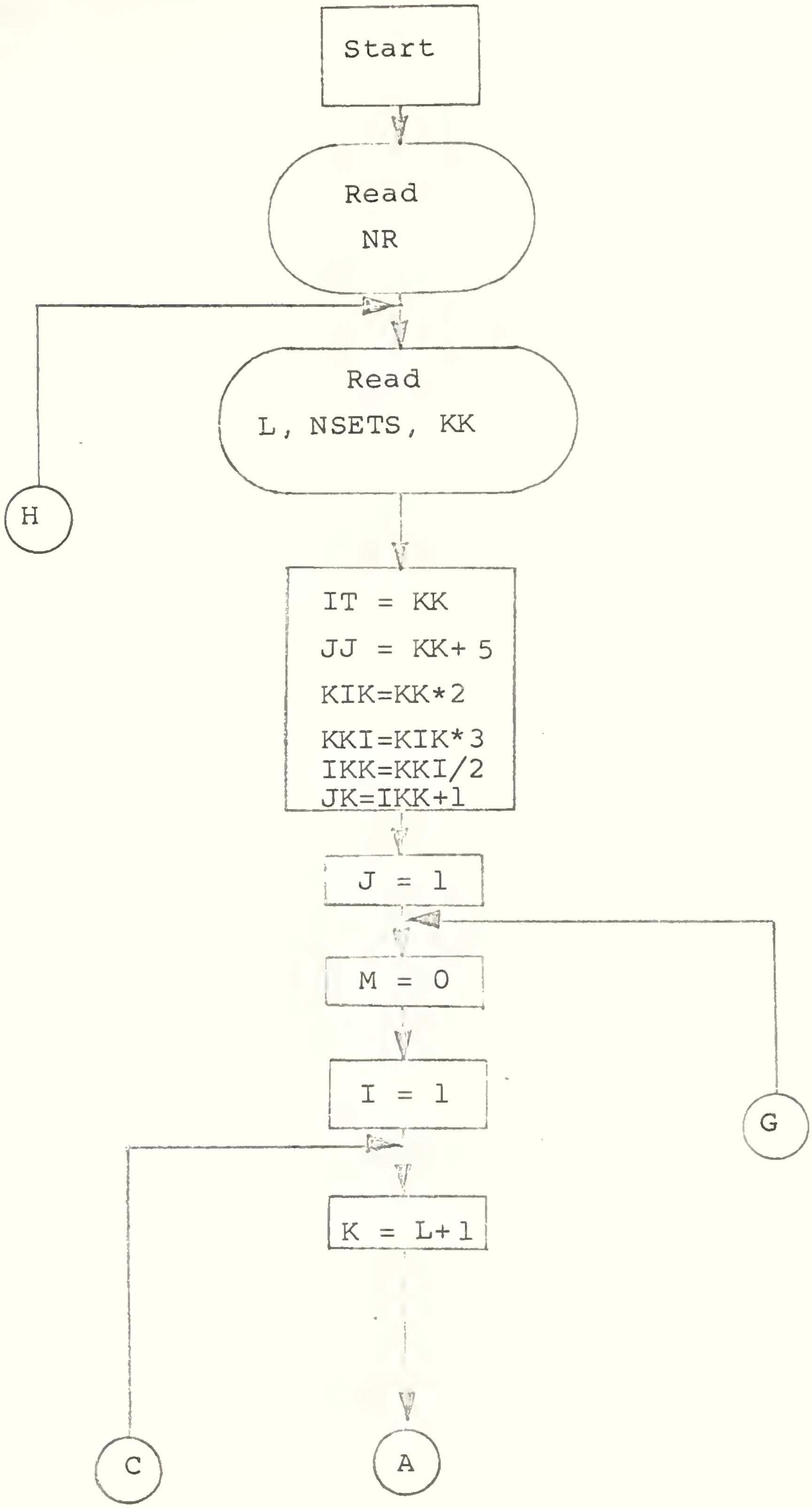
Certain nine digit numbers are chosen from the random number table and through various arithmetic operations are converted into simulated weight and value data, as well as one simulated maximum allowable weight load limit. The first random number chosen for conversion into the simulated data is determined by the input variable "I". Subsequent numbers from the random number table are chosen by a self-generating random device. The data may be produced as output in either floating or fixed point format, or both (see sense switch settings below).

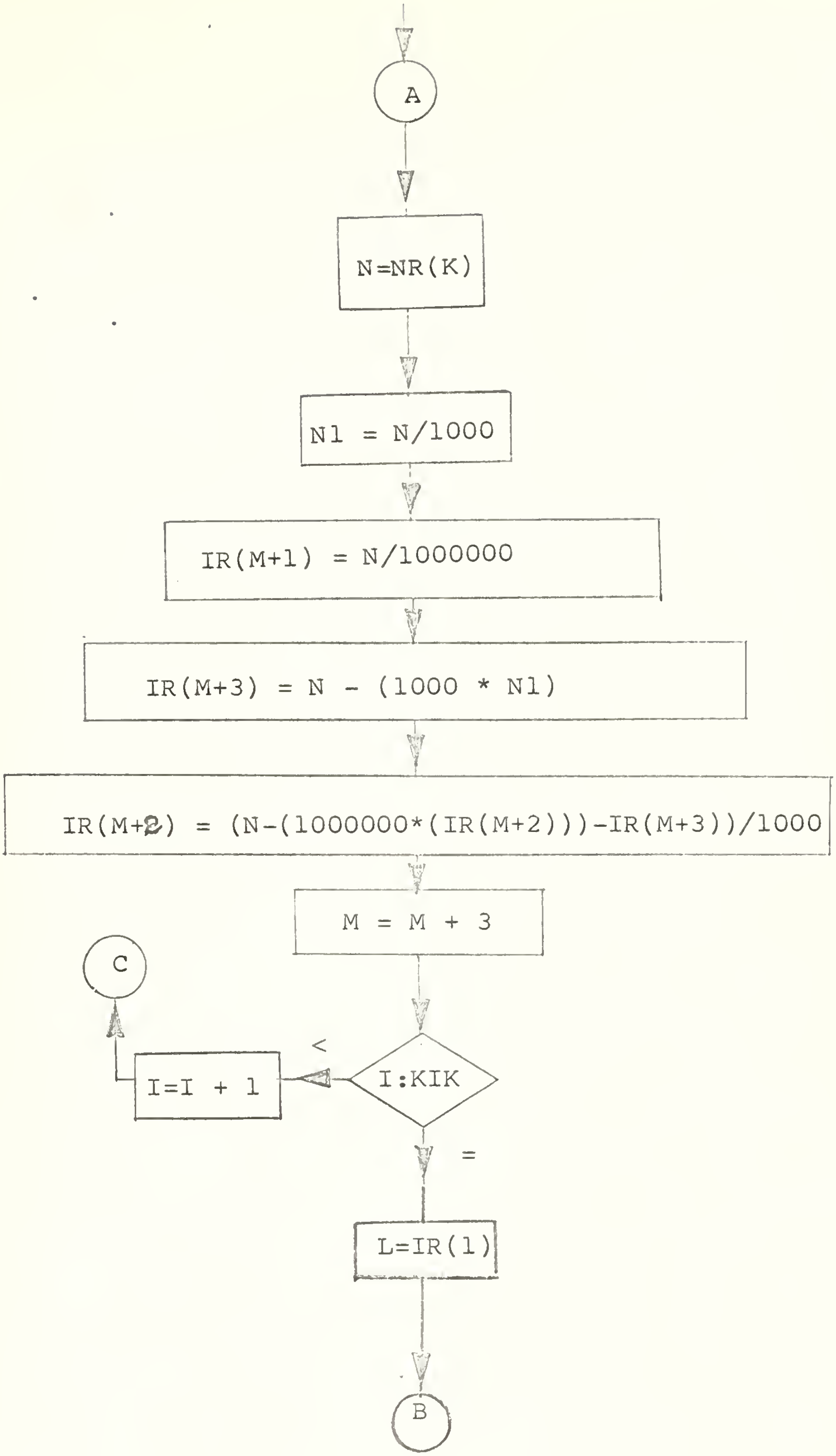
Sense Switch Settings

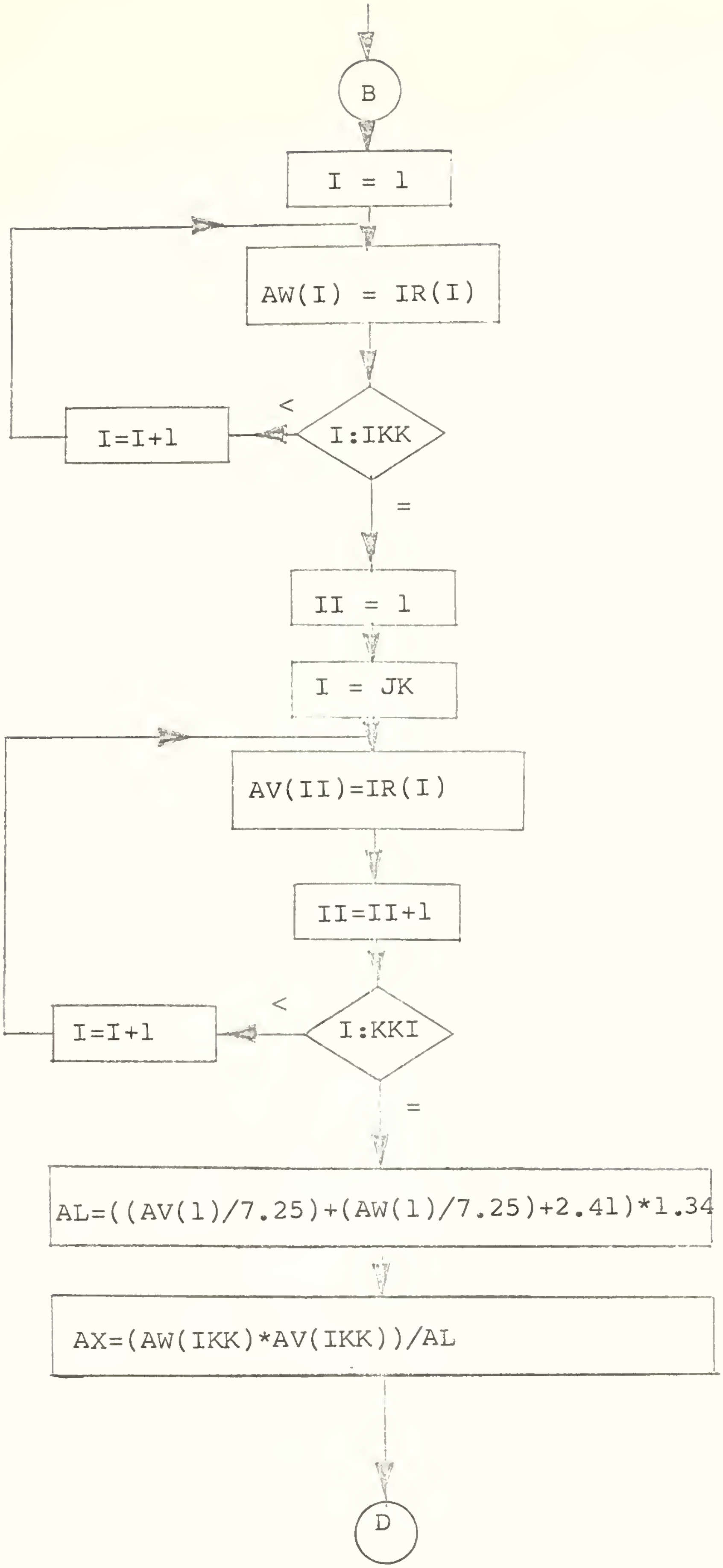
Sense Switch 1: When placed in the "on" position the output will consist of one punched card representing maximum allowable weight load limit, one card representing the number of data points to follow, and finally the appropriate (determined by the input variable "KK") number of punched cards containing both simulated weights and values.

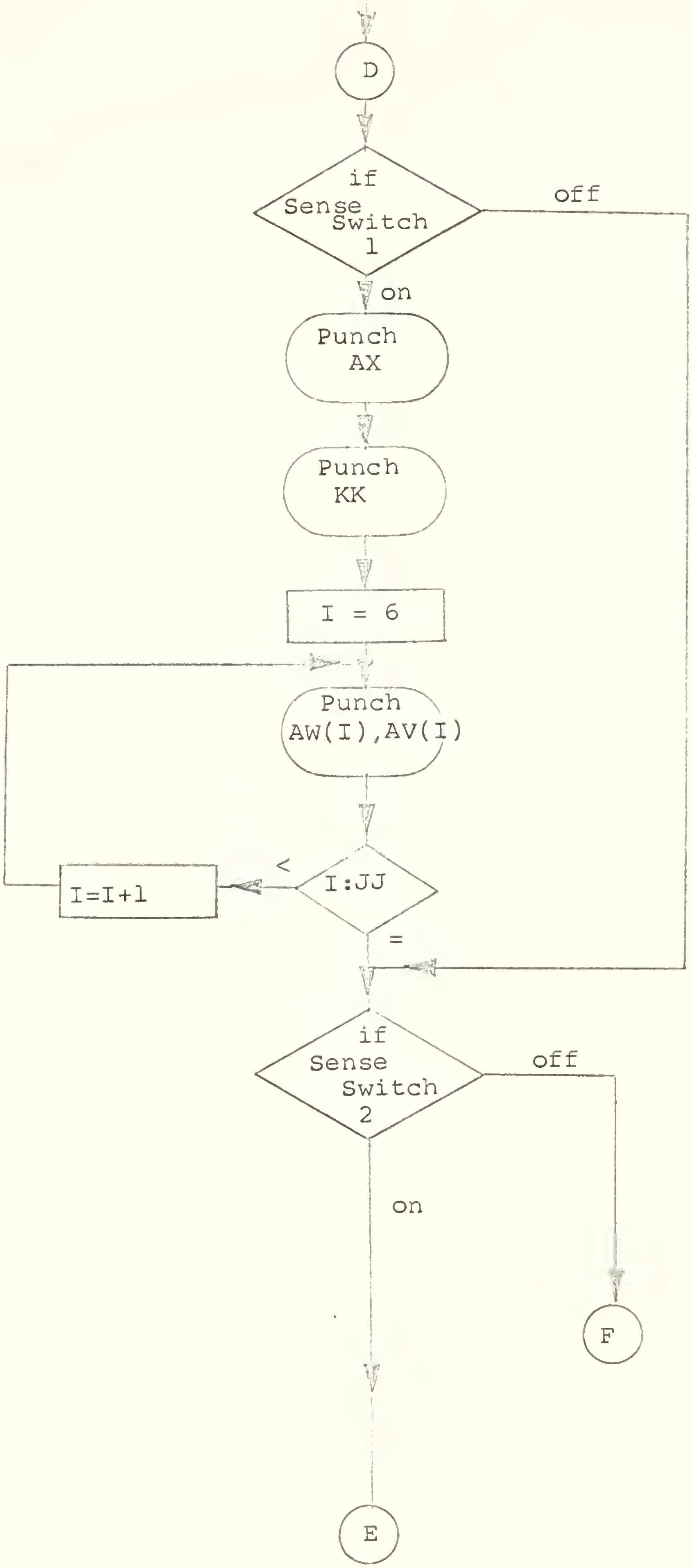
Sense Switch 2: When placed in the "on" position the output will consist of twenty-four (24) punched cards containing random, fixed point, numbers.

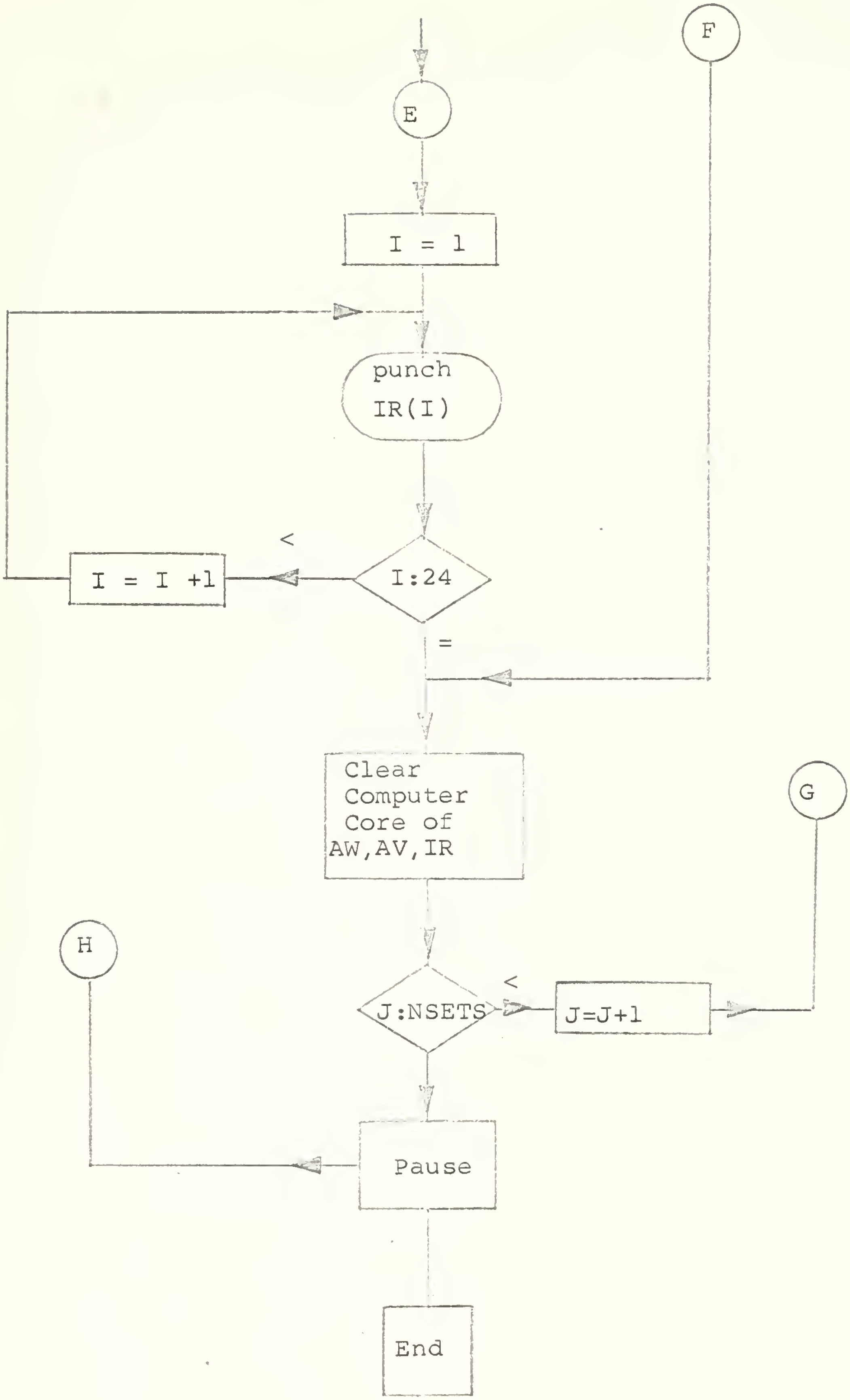
FLOW DIAGRAM FOR COMPUTER PROGRAM
TO GENERATE RANDOM DATA




A
N=NR(K)
N1 = N/1000
IR(M+1) = N/1000000
IR(M+3) = N - (1000 * N1)
IR(M+2) = (N-(1000000*(IR(M+2)))-IR(M+3))/1000
M = M + 3
C
I=I + 1
<
I:KIK
=
L=IR(1)
B


B
I = 1
AW(I) = IR(I)
<
I:IKK
I=I+1
=
II = 1
I = JK
AV(II)=IR(I)
II=II+1
<
I:KKI
I=I+1
=
AL=((AV(1)/7.25)+(AW(1)/7.25)+2.41)*1.34
AX=(AW(IKK)*AV(IKK))/AL
D


D
if
Sense
Switch
1
off
on
Punch
AX
Punch
KK
I = 6
Punch
AW(I),AV(I)
I:JJ
<
I=I+1
=
if
Sense
Switch
2
off
on
F
E


F
E
I = 1
punch
IR(I)
<
I = I +1
I:24
=
Clear
Computer
Core of
AW,AV,IR
G
H
<
J:NSETS
J=J+1
Pause
End

PROGRAM TO GENERATE RANDOM DATA

```
0839
      DIMENSION NR(1040), IR(800), AW(400), AV(400)
      READ 101,(NR(I),I = 1,1040)
    2 READ 102, L, NSETS, KK
  102 FORMAT (3I5)
  101 FORMAT(8I9)
      IT = KK
      JJ = KK + 5
      KIK = KK*2
      KKI = KIK*2
      IKK = KKI/2
      JK = IKK + 1
      DO 50 J = 1, NSETS
      M = 0
      DO 30 I = 1, KIK
      K = L + I
      N = NR(K)
      N1 = N/1000
      IR(M+1) = N/1000000
      IR(M+3) = N - (1000*N1)
      IR(M+2) = (N -(1000000 * (IR(M +1)))- IR(M+3))/1000
      M = M +3
   30 CONTINUE
      L = IR(1)
      DO 42 I = 1, IKK
      AW(I) = IR(I)
   42 CONTINUE
      II = 1
      DO 43 I = JK, KKI
      AV(II) = IR(I)
      II = II + 1
   43 CONTINUE
      AL =((AV(1)/7.25) + (AW(1)/7.25) + 1.41 )*1.34
      AX = (AW(IKK)*AV(IKK))/AL
      IF(SENSE SWITCH 1) 59, 52
   59 PUNCH 80, AX
   80 FORMAT(F20.8)
      PUNCH 81, KK
   81 FORMAT (I3)
      DO 53 I = 6, JJ
      PUNCH 54, AW(I), AV(I)
   54 FORMAT (2F20.8)
   53 CONTINUE
   52 IF(SENSE SWITCH 2) 55, 56
   55 DO 60 I = 1, 24
   61 FORMAT (I5)
   60 PUNCH 61, IR(I)
   56 CONTINUE
      DO 70 I = 1,800
   70 IR(I) = 0
      DO 71 I = 1,400
      AV(I) = 0.0
   71 AW(I) = 0.0
```

```
50 CONTINUE
   PAUSE
   GO TO 2
   END
```

COMPUTER PROGRAMS TO DETERMINE BY DIRECT ENUMERATION, THE OPTIMUM LOAD WEIGHT WITH MAXIMUM CARGO VALUE FOR SIX, TEN AND TWELVE ITEMS (OR PACKAGES)

Purpose

The purpose of these programs is to compute all possible weight and value combinations, for the appropriate number of data points (6, 10, or 12), and to determine the one set of data points having the greatest total value but with a total weight equal to or less than a specified maximum allowable weight load limit. Since these following three programs are identical except for varying numbers of input data points which are required, only one explanation is given here with exceptions being noted where necessary.

Language

Fortran II (IBM 1620 Computer).

Symbolic Dictionary

Variable	S/A*	I/O**	Description
WMAX	S	I&O	Maximum allowable load (or weight) limit of vehicle to be loaded.
NEN	S	I	Dummy variable. Equivalent to the number of items (or packages) to be considered for loading.

---

*S - Single variable; A - Array of variables

**I - Input; O - Output.

SUMWS	S	O	Total possible weight of cargo to be loaded.
SUMVS	S	O	Total maximum possible value of cargo to be loaded.
AW, AV BW, BV CW, CV DW, DV EW, EV GW, GV	S	I	Weights and values, respectively of six packages to be considered for loading.
PW, PV QW, QV RW, RV SW, SV	S	I	Additional (to the above six) weights and values, respectively of ten packages to be considered for loading.
TW, TV UW, UV	S	I	Additional (to the above ten) weights and values, respectively of twelve packages to be considered for loading.

<u>Program Routine</u>

The maximum total value of cargo, which has a total weight equal to or less than the maximum allowable load (or weight) limit is determined by making all possible summations of the possible combinations of package weights and all possible summations of the corresponding possible combinations of package values. The output is in a form which indicates the total maximum cargo loading weight, the maximum allowable load (or weight) limit, and the total maximum cargo value. In addition the output contains a loading schedule having a heading (of alphabetic letters A, B, C, etc.) corresponding to the individual data points

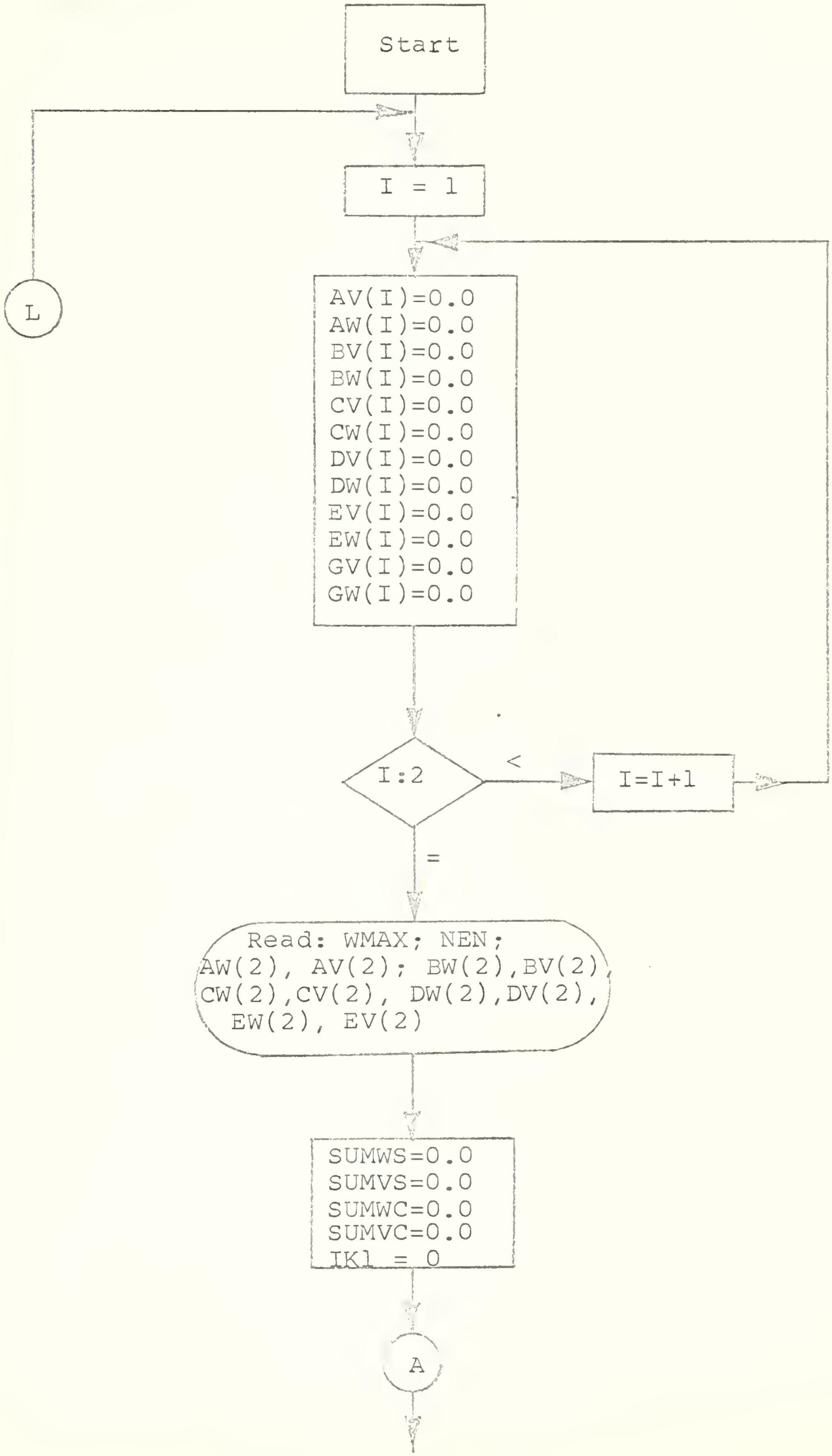
and either the number 1 or the number 2 underneath the respective alphabetic letter. The number 2 indicates that the package is to be loaded and is included in the loading schedule.

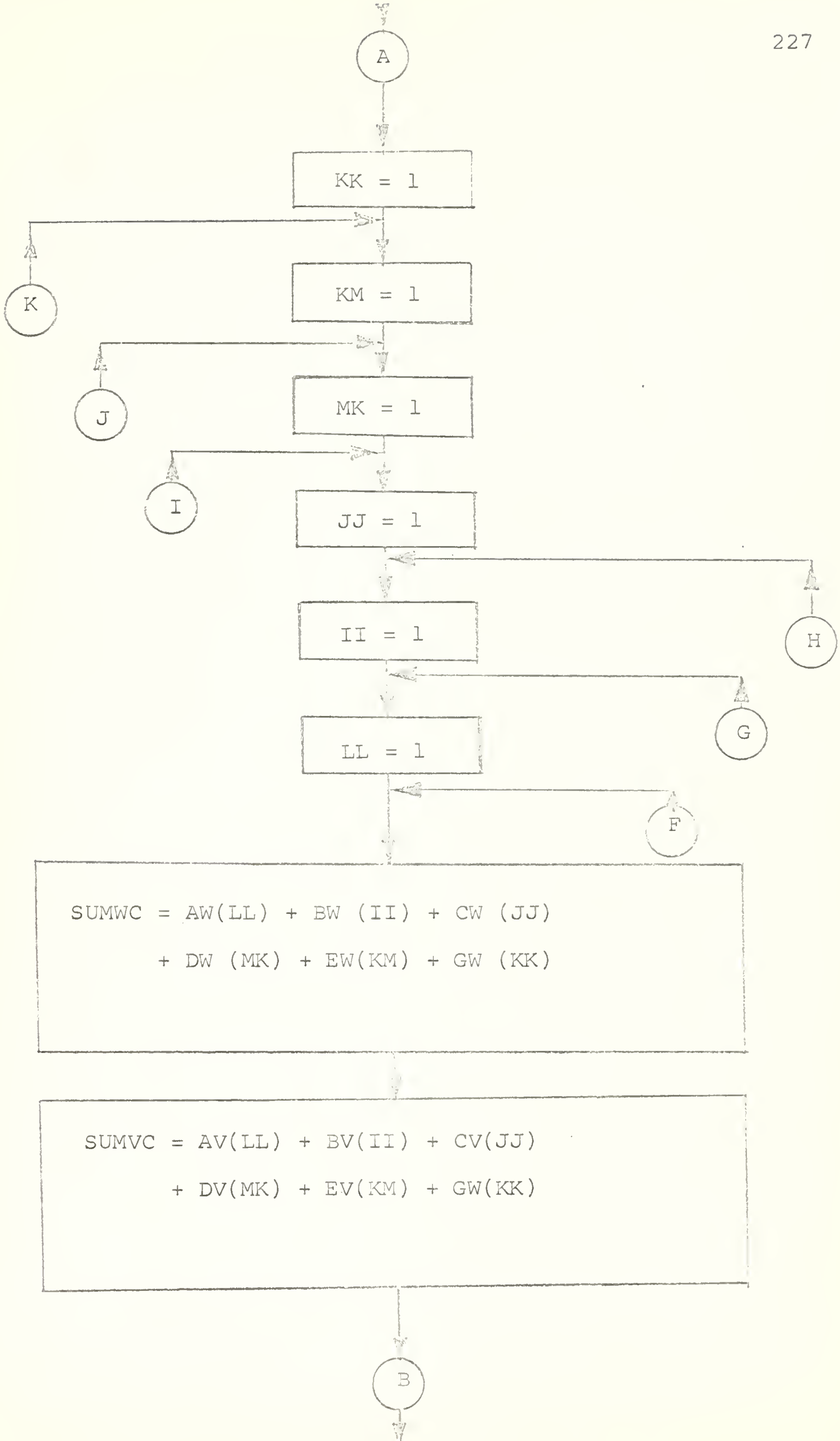
Sense Switch Settings

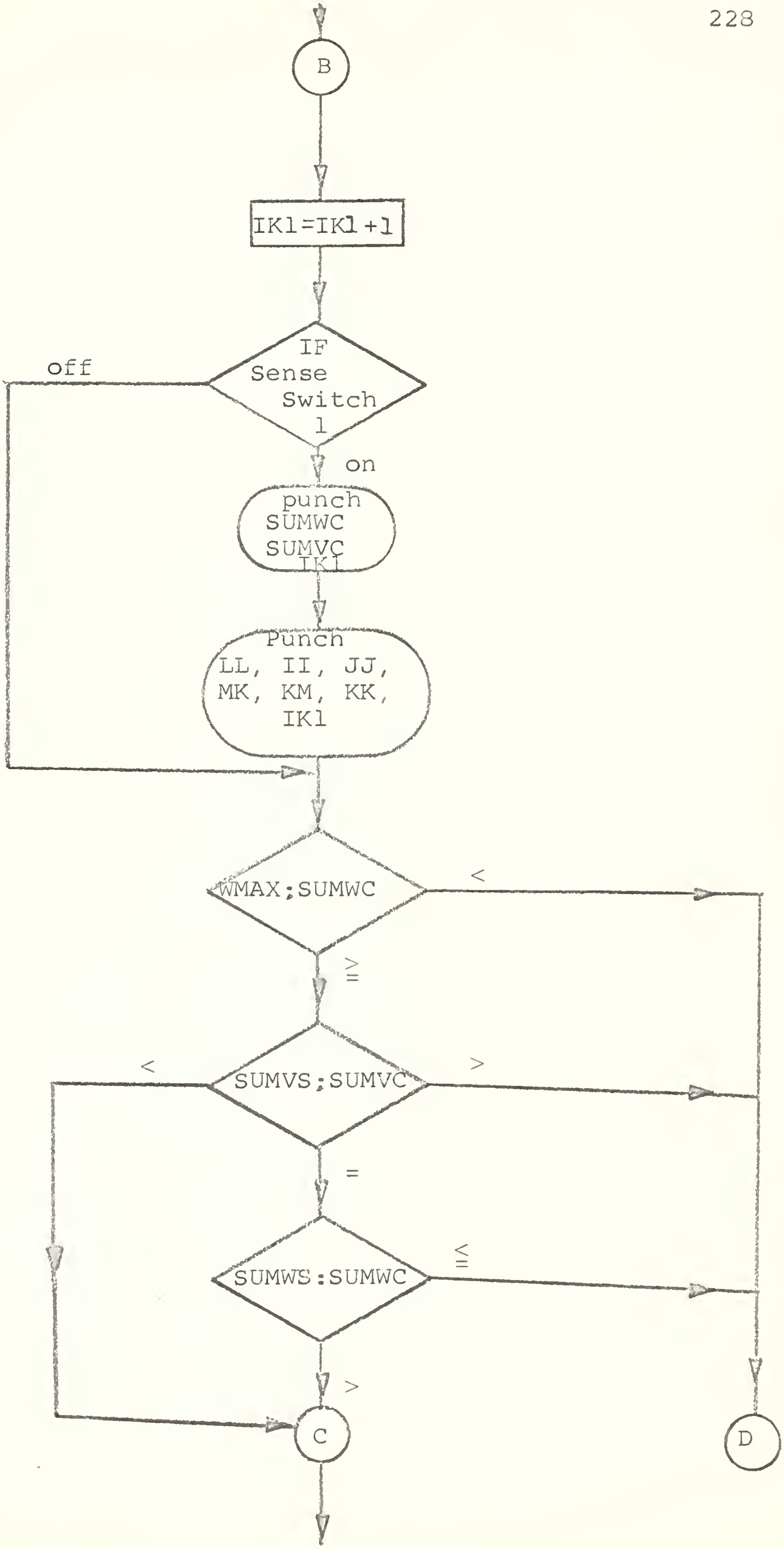
Sense Switch 1: When placed in the "on" position an output consisting of each individual calculation will be produced. The output will contain the combinations of packages being considered and the total combined weight and value of that computation.

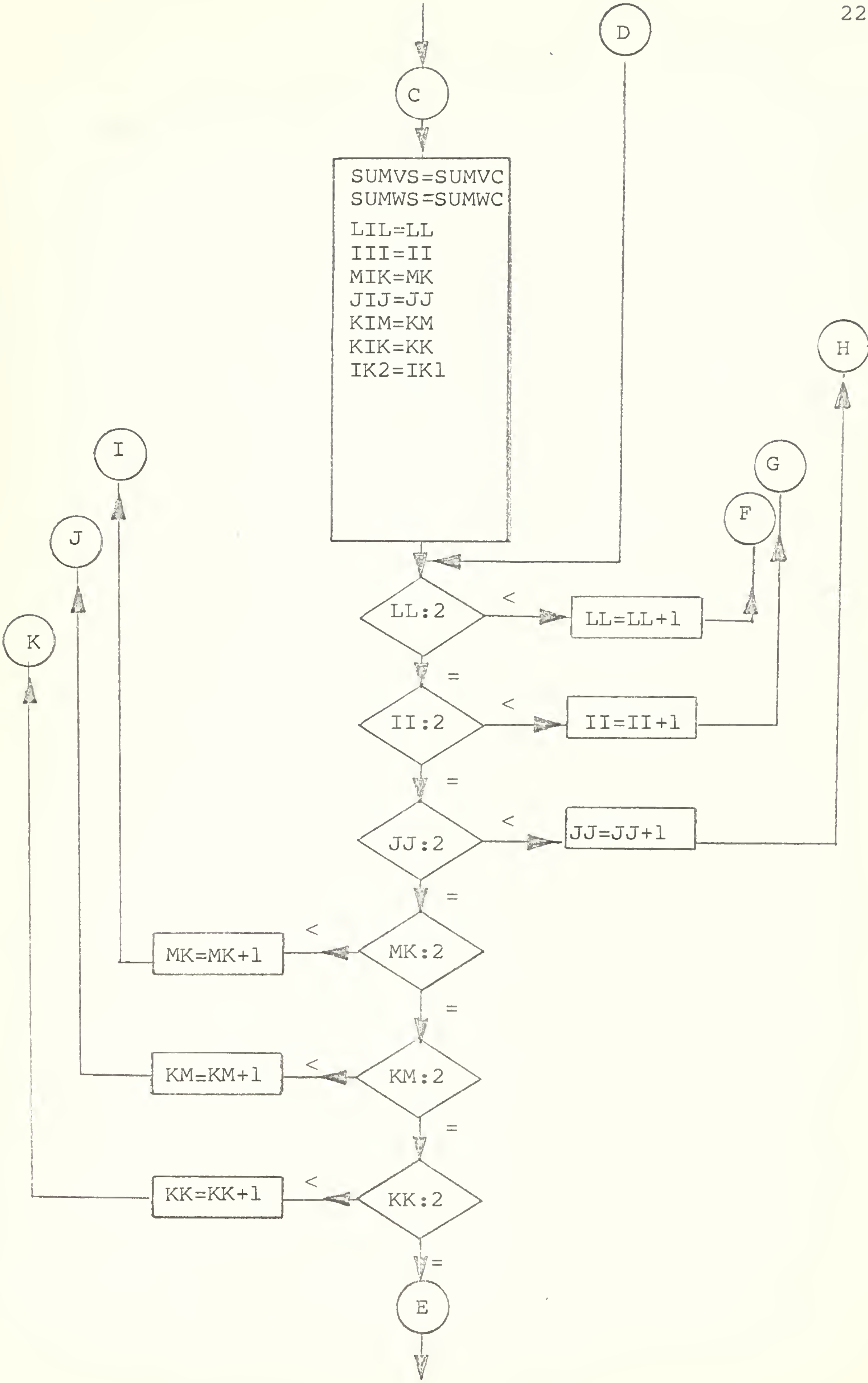
Sense Switch 3: When placed in the "on" position the program will pause at the completion of all computations. By pressing "start", on the computer console, the program will continue with another set of data. When sense switch 3 is in the 'off" position the program will continue with another set of data automatically, when completed with the previous computations.

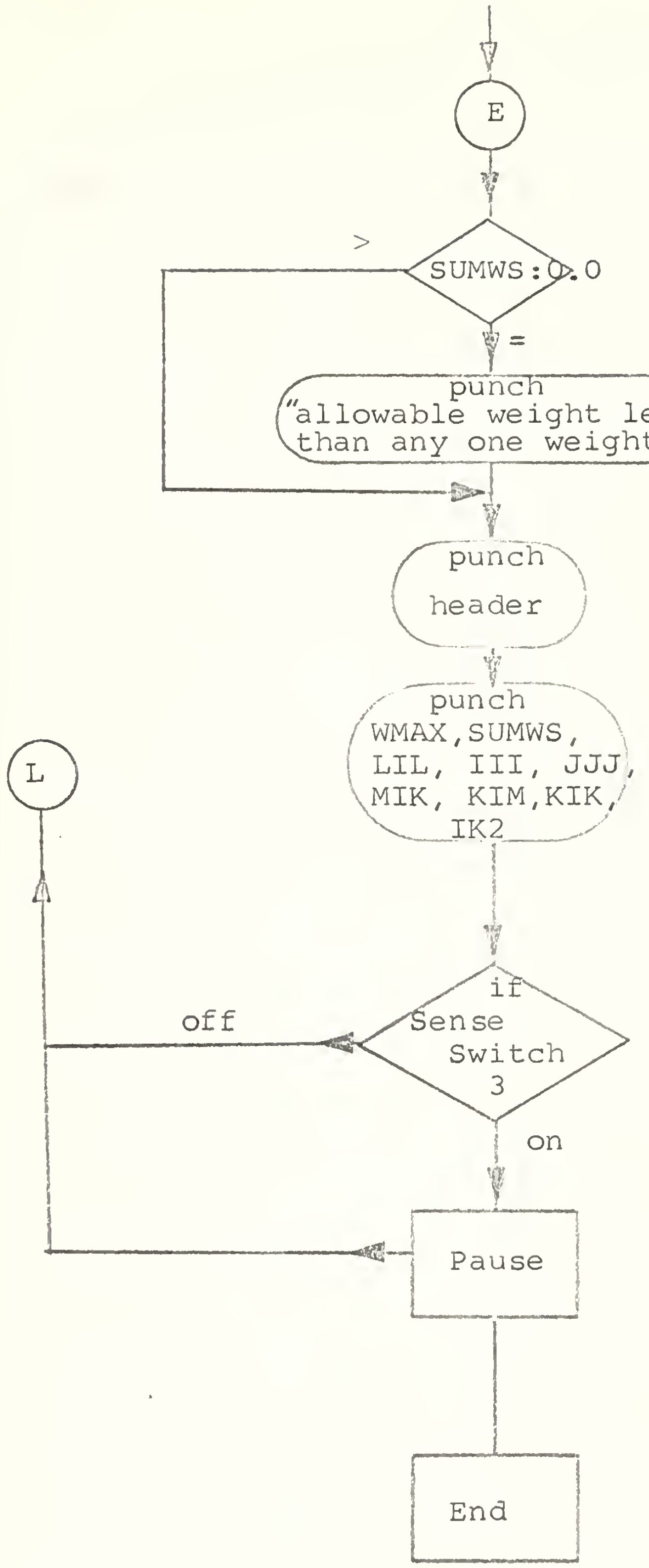
FLOW DIAGRAM FOR COMPUTER PROGRAM TO DETERMINE, BY DIRECT ENUMERATION, THE OPTIMUM LOAD WEIGHT WITH MAXIMUM CARGO VALUE FOR SIX ITEMS




A
KK = 1
K
KM = 1
J
MK = 1
I
JJ = 1
H
II = 1
G
LL = 1
F
SUMWC = AW(LL) + BW (II) + CW (JJ)
+ DW (MK) + EW(KM) + GW (KK)
SUMVC = AV(LL) + BV(II) + CV(JJ)
+ DV(MK) + EV(KM) + GW(KK)
B


B
IK1=IK1+1
IF Sense Switch 1
off
on
punch SUMWC SUMVC IK1
Punch LL, II, JJ, MK, KM, KK, IK1
WMAX;SUMWC
<
≧
SUMVS;SUMVC
<
>
=
SUMWS:SUMWC
≦
>
C
D


C
D
SUMVS=SUMVC
SUMWS=SUMWC
LIL=LL
III=II
MIK=MK
JIJ=JJ
KIM=KM
KIK=KK
IK2=IK1
H
G
F
I
J
K
LL:2
<
LL=LL+1
=
II:2
<
II=II+1
=
JJ:2
<
JJ=JJ+1
=
MK:2
<
MK=MK+1
=
KM:2
<
KM=KM+1
=
KK:2
<
KK=KK+1
=
E


E
SUMWS:0.0
>
=
punch
"allowable weight less than any one weight"
punch
header
punch
WMAX,SUMWS,
LIL, III, JJJ,
MIK, KIM,KIK,
IK2
L
if
Sense
Switch
3
off
on
Pause
End

```
C         PROGRAM TO DETERMINE, BY DIRECT ENUMERATION, THE OPTIMUM
C         LOAD WEIGHT WITH MAXIMUM CARGO VALUE FOR SIX ITEMS
      DIMENSION AW(2), BW(2), CW(2), DW(2), EW(2), GW(2)
      DIMENSION AV(2), BV(2), CV(2), DV(2), EV(2), GV(2)
    2 DO 10 I = 1,2
      AW(I) = 0.0
      AV(I) = 0.0
      BW(I) = 0.0
      BV(I) = 0.0
      CW(I) = 0.0
      CV(I) = 0.0
      DW(I) = 0.0
      DV(I) = 0.0
      EW(I) = 0.0
      EV(I) = 0.0
      GW(I) = 0.0
      GV(I) = 0.0
   10 CONTINUE
    3 FORMAT (F20.8)
    1 FORMAT (2F20.8)
    4 FORMAT(I3)
      READ 3, WMAX
      READ 4, MEN
      READ 1, AW(2), AV(2)
      READ 1, BW(2), BV(2)
      READ 1, CW(2), CV(2)
      READ 1, DW(2), DV(2)
      READ 1, EW(2), EV(2)
      READ 1, GW(2), GV(2)
      SUMWS = 0.0
      SUMVS = 0.0
      SUMWC = 0.0
      SUMVC = 0.0
      IK1 = 0
      DO 40 KK = 1,2
      DO 40 KM = 1,2
      DO 40 MK = 1,2
      DO 40 JJ = 1,2
      DO 40 II = 1,2
      DO 40 LL = 1,2
      SUMWC = AW(LL) + BW(II) + CW(JJ) + DW(MK) + EW(KM) + GW(KK)
      SUMVC = AV(LL) + BV(II) + CV(JJ) + DV(MK) + EV(KM) + GV(KK)
      IK1 = IK1 + 1
      IF(SENSE SWITCH 1) 37, 38
   37 PUNCH 57, SUMWC, SUMVC, IK1
   57 FORMAT(F20.8,5X,F20.8,25X,I5)
      PUNCH 53, LL,II,JJ,MK,KM,KK,IK1
   53 FORMAT(6I5,40X,I5)
   38 IF(WMAX- SUMWC) 40 , 39, 39
   39 IF(SUMVS - SUMVC) 69,68,40
   68 IF(SUMWS - SUMWC) 40, 40, 69
   69 SUMVS = SUMVC
      SUMWS = SUMWC
      LII = LL
```

```
      III = II
      MIK = MK
      JIJ = JJ
      KIM = KM
      KIK = KK
      IK2 = IK1
   40 CONTINUE
      IF(SUMWS) 81, 81, 80
   81 PUNCH 82
   82 FORMAT(41HALLOWABLE WEIGHT LESS THAN ANY ONE WEIGHT)
   80 PUNCH 50
   50 FORMAT(25H MAXIMUM ALLOWABLE WEIGHT,4X,24H MAXIMUM POSSIBLE WEIGHT
  501)
      PUNCH 51, WMAX, SUMWS
   51 FORMAT(F20.8,4X,F20.8)
      PUNCH 52
      PUNCH 53, LIL, III,JIJ, MIK, KIM, KIK,IK2
      PUNCH 66, SUMVS
   66 FORMAT(16H MAXIMUM VALUE =,1X,F20.8)
   52 FORMAT(3X,2H A,3X,2H B,3X,2H C,3X,2H D,3X,2H E,3X,2H G)
      IF(SENSE SWITCH 3) 98, 99
   98 PAUSE
   99 GO TO 2
      END
```

```
C         PROGRAM TO DETERMINE, BY DIRECT ENUMERATION, THE OPTIMUM
C         LOAD WEIGHT WITH MAXIMUM CARGO VALUE FOR TEN ITEMS
      DIMENSION AW(2), BW(2), CW(2), DW(2), EW(2), GW(2), PW(2), QW(2)
      DIMENSION RW(2), SW(2), AV(2), BV(2), CV(2), DV(2),EV(2), GV(2)
      DIMENSION PV(2), QV(2), RV(2), SV(2)
    2 DO 10 I = 1,2
      AW(I) = 0.0
      AV(I) = 0.0
      BW(I) = 0.0
      BV(I) = 0.0
      CW(I) = 0.0
      CV(I) = 0.0
      DW(I) = 0.0
      DV(I) = 0.0
      EW(I) = 0.0
      EV(I) = 0.0
      GW(I) = 0.0
      GV(I) = 0.0
      PW(I) = 0.0
      PV(I) = 0.0
      QW(I) = 0.0
      QV(I) = 0.0
      RW(I) = 0.0
      RV(I) = 0.0
      SW(I) = 0.0
      SV(I) = 0.0
   10 CONTINUE
    3 FORMAT (F20.8)
    1 FORMAT (2F20.8)
    4 FORMAT(I3)
      READ 3, WMAX
      READ 4, NEN
      READ 1, AW(2), AV(2)
      READ 1, BW(2), BV(2)
      READ 1, CW(2), CV(2)
      READ 1, DW(2), DV(2)
      READ 1, EW(2), EV(2)
      READ 1, GW(2), GV(2)
      READ 1, PW(2), PV(2)
      READ 1, QW(2), QV(2)
      READ 1, RW(2), RV(2)
      READ 1, SW(2), SV(2)
      SUMWS = 0.0
      SUMVS = 0.0
      SUMWC = 0.0
      SUMVC = 0.0
      IK1 = 0
   42 DO 40 KL = 1,2
      DO 40 LK = 1,2
      DO 40 ML = 1,2
      DO 40 LM = 1,2
      DO 40 KK = 1,2
      DO 40 KM = 1,2
      DO 40 MK = 1,2
```

```
      DO 40 JJ = 1,2
      DO 40 II = 1,2
      DO 40 LL = 1,2
      SUMWC = AW(LL) + BW(II) + CW(JJ) + DW(MK) + EW(KM) + GW(KK) +
     1 PW(LM) + QW(ML) + RW(LK) + SW(KL)
      SUMVC = AV(LL) + BV(II) + CV(JJ) + DV(MK) + EV(KM) + GV(KK) +
     1 PV(LM) + QV(ML) + RV(LK) + SV(KL)
      IK1 = IK1 + 1
      IF(SENSE SWITCH 1) 37, 38
   37 PUNCH 57, SUMWC, SUMVC, IK1
   57 FORMAT(F20.8,5X,F20.8,25X,I5)
      PUNCH 53, LL,II,JJ,MK,KM,KK,LM,ML,LK,KL,IK1
   53 FORMAT(10I5,20X,I5)
   38 IF(WMAX- SUMWC) 40, 39, 39
   39 IF(SUMVS - SUMVC) 69,68,40
   68 IF(SUMWS - SUMWC) 40, 40, 69
   69 SUMVS = SUMVC
      SUMWS = SUMWC
      LIL = LL
      III = II
      MIK = MK
      JIJ = JJ
      KIM = KM
      LIM = LM
      KIK = KK
      MIL = ML
      LIK = LK
      KIL = KL
      IK2 = IK1
   40 CONTINUE
      IF(SUMWS) 81, 81, 80
   81 PUNCH 82
   82 FORMAT(41HALLOWABLE WEIGHT LESS THAN ANY ONE WEIGHT)
   80 PUNCH 50
   50 FORMAT(25H MAXIMUM ALLOWABLE WEIGHT,4X,24H MAXIMUM POSSIBLE WEIGHT
  501)
      PUNCH 51, WMAX,SUMWS
   51 FORMAT(F20.8,4X,F20.8)
      PUNCH 52
      PUNCH 53, LIL, III,JIJ,MIK,KIM,KIK,LIM,MIL,LIK,KIL, IK2
      PUNCH 66, SUMVS
   66 FORMAT(16H MAXIMUM VALUE =,1X,F20.8)
   52 FORMAT(3X,2H A,3X,2H B,3X,2H C,3X,2H D,3X,2H E,3X,2H G,3X,2H P,
     13X,2H Q, 3X,2H R,3X,2H S)
      IF(SENSE SWITCH 3) 98, 99
   98 PAUSE
   99 GO TO 2
      END
```

```
C          PROGRAM TO DETERMINE, BY DIRECT ENUMERATION, THE OPTIMUM
C          LOAD WEIGHT WITH MAXIMUM CARGO VALUE FOR TWELVE ITEMS
        DIMENSION AW(2), BW(2), CW(2), DW(2), EW(2), GW(2), PW(2), QW(2)
        DIMENSION RW(2), SW(2), AV(2), BV(2), CV(2), DV(2), EV(2), GV(2)
        DIMENSION PV(2), QV(2), RV(2), SV(2), TW(2), TV(2), UW(2), UV(2)
      2 DO 10 I = 1,2
        AW(I) = 0.0
        AV(I) = 0.0
        BW(I) = 0.0
        BV(I) = 0.0
        CW(I) = 0.0
        CV(I) = 0.0
        DW(I) = 0.0
        DV(I) = 0.0
        EW(I) = 0.0
        EV(I) = 0.0
        GW(I) = 0.0
        GV(I) = 0.0
        PW(I) = 0.0
        PV(I) = 0.0
        QW(I) = 0.0
        QV(I) = 0.0
        RW(I) = 0.0
        RV(I) = 0.0
        SW(I) = 0.0
        SV(I) = 0.0
        TW(I) = 0.0
        TV(I) = 0.0
        UW(I) =0.0
        UV(I) =0.0
     10 CONTINUE
      3 FORMAT (F20.8)
      1 FORMAT (2F20.8)
      4 FORMAT(I3)
        READ 3, WMAX
        READ 4, NEN
        READ 1, AW(2), AV(2)
        READ 1, BW(2), BV(2)
        READ 1, CW(2), CV(2)
        READ 1, DW(2), DV(2)
        READ 1, EW(2), EV(2)
        READ 1, GW(2), GV(2)
        READ 1, PW(2), PV(2)
        READ 1, QW(2), QV(2)
        READ 1, RW(2), RV(2)
        READ 1, SW(2), SV(2)
        READ 1, TW(2), TV(2)
        READ 1, UW(2), UV(2)
        SUMWS = 0.0
        SUMVS = 0.0
        SUMWC = 0.0
        SUMVC = 0.0
        IK1 = 0
     42 DO 40 IL = 1,2
```

```
      DO 40 LI = 1,2
      DO 40 KL = 1,2
      DO 40 LK = 1,2
      DO 40 ML = 1,2
      DO 40 LM = 1,2
      DO 40 KK = 1,2
      DO 40 KM = 1,2
      DO 40 MK = 1,2
      DO 40 JJ = 1,2
      DO 40 II = 1,2
      DO 40 LL = 1,2
      SUMWC = AW(LL) + BW(II) + CW(JJ) + DW(MK) + EW(KM) + GW(KK) +
     1 PW(LM) + QW(ML) + RW(LK) + SW(KL) + TW(LI) + UW(IL)
      SUMVC = AV(LL) + BV(II) + CV(JJ) + DV(MK) + EV(KM) + GV(KK) +
     1 PV(LM) + QV(ML) + RV(LK) + SV(KL) + TV(LI) + UV(IL)
      IK1 = IK1 + 1
      IF(SENSE SWITCH 1) 37, 38
   37 PUNCH 57, SUMWC, SUMVC, IK1
   57 FORMAT(F20.8,5X,F20.8,25X,I5)
      PUNCH 53, LL,II,JJ,MK,KM,KK,LM,ML,LK,KL,LI, IL, IK1
   53 FORMAT(12I5, 10X, I5)
   38 IF(WMAX- SUMWC) 40, 39, 39
   39 IF(SUMVS - SUMVC) 69,68,40
   68 IF(SUMWS - SUMWC) 40, 40, 69
   69 SUMVS = SUMVC
      SUMWS = SUMWC
      LIL = LL
      III = II
      MIK = MK
      JIJ = JJ
      KIM = KM
      LIM = LM
      KIK = KK
      MIL = ML
      LIK = LK
      KIL = KL
      LII = LI
      IIL = IL
      IK2 = IK1
   40 CONTINUE
      IF(SUMWS) 81, 81, 80
   81 PUNCH 82
   82 FORMAT(41HALLOWABLE WEIGHT LESS THAN ANY ONE WEIGHT)
   80 PUNCH 50
   50 FORMAT(25H MAXIMUM ALLOWABLE WEIGHT,4X,24H MAXIMUM POSSIBLE WEIGHT
     501)
      PUNCH 51, WMAX,SUMWS
   51 FORMAT(F20.8,4X,F20.8)
      PUNCH 52
      PUNCH 53, LIL, III,JIJ,MIK,KIM,KIK,LIM,MIL,LIK,KIL, LII , IIL, IK2
      PUNCH 66, SUMVS
   66 FORMAT(16H MAXIMUM VALUE =,1X,F20.8)
   52 FORMAT(3X,2H A,3X,2H B,3X,2H C,3X,2H D,3X,2H E,3X,2H G,3X,2H P,
     13X,2H Q, 3X,2H R,3X,2H S,3X,2H T,3X,2H U)
```

```
      IF(SENSE SWITCH 3) 98, 99
   98 PAUSE
   99 GO TO 2
      END
```

APPENDIX II

ALL POSSIBLE COMBINATIONS OF THE GIVEN DATA ($2^n = 2^6 = 64$)

GIVEN DATA:	1	2	3	4	5	6	TOTAL WEIGHT	TOTAL VALUE	TOTAL CUBE
WEIGHT	100.00	1.00	0.50	451.237	24.563	88.511			
VALUE	1000.00	500.00	100.00	500.00	88.88	217.83			
CUBE	0.50	0.10	0.01	2.00	1.00	1.50			
LINE NO.									
1							0.0	0.0	0.0
2	x						100.00	1000.00	0.50
3		x					1.00	500.00	0.10
4			x				0.50	100.00	0.01
5				x			451.237	500.00	2.00
6					x		24.563	88.88	1.00
7						x	88.511	217.83	1.50
8	x	x					101.00	1500.00	0.60
9	x		x				100.5	1100.00	0.51
10	x			x			551.237	1500.00	2.50
11	x				x		124.563	1088.88	1.50
12	x					x	188.511	1217.83	2.00
13		x	x				1.50	600.00	0.11
14		x		x			452.237	1000.00	2.10
15		x			x		25.563	588.88	1.10
16		x				x	89.511	717.83	1.60
17			x	x			451.737	600.00	2.01
18			x		x		25.063	188.88	1.01
19			x			x	89.011	317.83	1.51
20				x	x		475.800	588.88	3.00
21				x		x	539.748	717.83	3.50
22					x	x	113.074	306.71	2.50
23	x	x	x				101.500	1600.00	0.61
24	x	x		x			552.237	2000.00	2.60
25	x	x			x		125.563	1588.88	1.60
26	x	x				x	189.511	1717.83	2.10
27	x		x	x			551.737	1600.00	2.51
28	x		x		x		125.063	1188.88	1.51
29	x		x			x	189.011	1317.83	2.01

GIVEN DATA:	1	2	3	4	5	6	TOTAL WEIGHT	TOTAL VALUE	TOTAL CUBE
WEIGHT	100.00	1.00	0.50	451.237	24.563	88.511			
VALUE	1000.00	500.00	100.00	500.00	88.88	217.83			
CUBE	0.50	0.10	0.01	2.00	1.00	1.50			
LINE NO.									
30	x			x	x		575.800	1588.88	3.50
31	x			x		x	639.748	1717.83	4.00
32	x				x	x	213.074	1306.71	3.00
33		x	x	x			452.737	1100.00	2.11
34		x	x		x		25.063	688.88	1.11
35		x	x			x	90.011	817.83	1.61
36		x		x	x		476.800	1088.88	3.10
37		x		x		x	540.748	1217.83	3.60
38		x			x	x	114.074	806.71	2.60
39			x	x	x		476.300	688.88	3.01
40			x	x		x	540.248	817.83	3.51
41			x		x	x	113.574	406.71	2.51
42				x	x	x	564.311	806.71	4.50
43	x	x	x	x			552.737	2100.00	2.61
44	x	x	x		x		126.063	1688.88	1.61
45	x	x	x			x	190.011	1817.83	2.11
46	x	x		x	x		576.80	2088.88	3.60
47	x	x		x		x	640.748	2217.83	4.10
48	x	x			x	x	214.074	1806.71	3.10
49	x		x	x	x		576.300	1688.88	3.51
50	x		x	x		x	640.248	1817.83	4.01
51	x		x		x	x	213.574	1406.71	3.01
52	x			x	x	x	664.311	1806.71	4.00
53		x	x	x	x		477.300	1188.88	3.11
54		x	x	x		x	541.248	1317.83	3.61
55		x	x		x	x	114.574	906.71	2.61
56		x		x	x	x	565.311	1306.71	4.60
57			x	x	x	x	564.811	906.71	4.51
58	x	x	x	x	x		577.300	2188.88	3.61
59	x	x	x	x		x	641.248	2317.83	4.11
60	x	x	x		x	x	214.574	1906.71	3.11
61	x	x		x	x	x	665.311	2306.71	5.10
62	x		x	x	x	x	664.811	1906.71	5.01
63		x	x	x	x	x	565.811	1406.71	4.61
64	x	x	x	x	x	x	665.811	2406.71	5.11

OPTIMUM VALUES FROM ALL POSSIBLE VALUES OF THE GIVEN DATA

LINE NO.	WEIGHT TOTAL	VALUE TOTAL	CUBE TOTAL
64	665.811	$ 2406.71	5.11
59	641.248	2317.83	4.11
47	640.748	2217.83	4.10
58	577.300	2188.88	3.61
43	552.737	2100.00	2.61
24	552.237	2000.00	2.60
60	214.574	1906.71	3.11
45	190.110	1817.83	2.11
26	189.511	1717.83	2.10
44	126.063	1688.88	1.61
23	101.500	1600.00	0.61
8	101.000	1500.00	0.60
9	100.500	1100.00	0.51
2	100.000	1000.00	0.50
35	90.011	817.83	1.61
16	89.511	717.83	1.60
34	25.063	688.88	1.11
13	1.500	600.00	0.11
3	1.000	500.00	0.10
4	0.500	100.00	0.01
1	0.000	000.00	0.00

EXAMPLE PROBLEMS UTILIZING PROGRAM TO COMPUTE APPROXIMATE OPTIMUM WEIGHT
WITH MAXIMUM VALUE

INPUT DATA

PROBLEM NO. 1
191.0
6

24.563	88.88
451.237	500.0
0.5	100.00
88.511	217.83
1.0	500.00
100.0	1000.00

PROBLEM NO. 2
639.9
6

24.563	88.88
451.237	500.0
0.5	100.00
88.511	217.83
1.0	500.00
100.0	1000.00

PROBLEM NO. 3
578.0
6

24.563	88.88
451.237	500.0
0.5	100.00
88.511	217.83
1.0	500.00
100.0	1000.00

PROBLEM NO. 4
552.5
6

24.563	88.88
451.237	500.0
0.5	100.00
88.511	217.83
1.0	500.00
100.0	1000.00

PROBLEM NO. 5
553.0
6

24.563	88.88
451.237	500.0
0.5	100.00
88.511	217.83
1.0	500.00
100.0	1000.00

SOLUTION

BEGINNING OF PROBLEM 1

FIRST SORT BY DESCENDING INDEX

INDEX	QUANTITY	VALUE
500.00000000	1.00000000	500.00000000
200.00000000	.50000000	100.00000000
10.00000000	100.00000000	1000.00000000
3.61845050	24.56300000	88.88000000
2.46105000	88.51100000	217.83000000
1.10806510	451.23700000	500.00000000

SECOND SORT BY DESCENDING QUANTITY HAVING IDENTICAL INDEXES

INDEX	QUANTITY	VALUE
500.00000000	1.00000000	500.00000000
200.00000000	.50000000	100.00000000
10.00000000	100.00000000	1000.00000000
3.61845050	24.56300000	88.88000000
2.46105000	88.51100000	217.83000000
1.10806510	451.23700000	500.00000000

LOADING SCHEDULE

QUANTITY	VALUE	INDEX
0.00000000	0.00000000	3.61845050
.50000000	100.00000000	200.00000000
1.00000000	500.00000000	500.00000000
100.00000000	1000.00000000	10.00000000
88.51100000	217.83000000	2.46105000

QUANTITY MAX	VALUE MAX	OBJECTIVE
190.01100000	1817.83000000	191.00000000

END OF PROBLEM 1

BEGINNING OF PROBLEM 2

FIRST SORT BY DESCENDING INDEX

INDEX	QUANTITY	VALUE
500.00000000	1.00000000	500.00000000
200.00000000	.50000000	100.00000000
10.00000000	100.00000000	1000.00000000
3.61845050	24.56300000	88.88000000
2.46105000	88.51100000	217.83000000
1.10806510	451.23700000	500.00000000

SECOND SORT BY DESCENDING QUANTITY HAVING IDENTICAL INDEXES

INDEX	QUANTITY	VALUE

500.00000000	1.00000000	500.00000000
200.00000000	.50000000	100.00000000
10.00000000	100.00000000	1000.00000000
3.61845050	24.56300000	88.88000000
2.46105000	88.51100000	217.83000000
1.10806510	451.23700000	500.00000000

LOADING SCHEDULE

QUANTITY	VALUE	INDEX
0.00000000	0.00000000	2.46105000
24.56300000	88.88000000	3.61845050
.50000000	100.00000000	200.00000000
1.00000000	500.00000000	500.00000000
100.00000000	1000.00000000	10.00000000
451.23700000	500.00000000	1.10806510

QUANTITY MAX	VALUE MAX	OBJECTIVE
577.30000000	2188.88000000	639.50000000

END OF PROBLEM 2

BEGINNING OF PROBLEM 3

FIRST SORT BY DESCENDING INDEX

INDEX	QUANTITY	VALUE
500.00000000	1.00000000	500.00000000
200.00000000	.50000000	100.00000000
10.00000000	100.00000000	1000.00000000
3.61845050	24.56300000	88.88000000
2.46105000	88.51100000	217.83000000
1.10806510	451.23700000	500.00000000

SECOND SORT BY DESCENDING QUANTITY HAVING IDENTICAL INDEXES

INDEX	QUANTITY	VALUE
500.00000000	1.00000000	500.00000000
200.00000000	.50000000	100.00000000
10.00000000	100.00000000	1000.00000000
3.61845050	24.56300000	88.88000000
2.46105000	88.51100000	217.83000000
1.10806510	451.23700000	500.00000000

LOADING SCHEDULE

QUANTITY	VALUE	INDEX
0.00000000	0.00000000	2.46105000
24.56300000	88.88000000	3.61845050
.50000000	100.00000000	200.00000000
1.00000000	500.00000000	500.00000000
100.00000000	1000.00000000	10.00000000
451.23700000	500.00000000	1.10806510

QUANTITY MAX	VALUE MAX	OBJECTIVE
577.30000000	[illegible].88000000	578.[illegible]

END OF PROBLEM 3

BEGINNING OF PROBLEM 4

FIRST SORT BY DESCENDING INDEX

INDEX	QUANTITY	VALUE
500.00000000	1.00000000	500.00000000
200.00000000	.50000000	100.00000000
10.00000000	100.00000000	1000.00000000
3.61845050	24.56300000	88.88000000
2.46105000	88.51100000	217.83000000
1.10806510	451.23700000	500.00000000

SECOND SORT BY DESCENDING QUANTITY HAVING IDENTICAL INDEXES

INDEX	QUANTITY	VALUE
500.00000000	1.00000000	500.00000000
200.00000000	.50000000	100.00000000
10.00000000	100.00000000	1000.00000000
3.61845050	24.56300000	88.88000000
2.46105000	88.51100000	217.83000000
1.10806510	451.23700000	500.00000000

LOADING SCHEDULE

QUANTITY	VALUE	INDEX
0.00000000	0.00000000	200.00000000
0.00000000	0.00000000	3.61845050
0.00000000	0.00000000	2.46105000
1.00000000	5[illegible].[illegible]	500.00000000
100.00000000	1[illegible].[illegible]	10.00000000
451.23700000	5[illegible].[illegible]	1.10806510

QUANTITY MAX	VALUE MAX	OBJECTIVE
552.237[illegible]	[illegible]	552.5[illegible]

END OF PROBLEM 4

BEGINNING OF PROBLEM 5

FIRST SORT BY DESCENDING INDEX

INDEX	QUANTITY	VALUE
500.[illegible]	1.[illegible]	5[illegible].[illegible]
200.[illegible]	.50[illegible]	10[illegible].[illegible]
10.00000000	1[illegible].[illegible]	10[illegible].[illegible]
3.6184[illegible]	24.56[illegible]	88.88[illegible]

2.461[illegible]	88.5110000[illegible]	217.8[illegible]
1.10806510	451.237[illegible]	5[illegible]

SECOND SORT BY DESCENDING QUANTITY HAVING IDENTICAL INDEXES

INDEX	QUANTITY	VALUE
500.00000000	1.[illegible]	5[illegible]
200.00000000	.50[illegible]	1[illegible]
10.00000000	10[illegible]	1[illegible]
3.61845050	24.563[illegible]	88.88[illegible]
2.46105000	88.511[illegible]	217.83[illegible]
1.10806510	451.237[illegible]	5[illegible]

LOADING SCHEDULE

QUANTITY	VALUE	INDEX
0.[illegible]	[illegible]	3.61845050
0.000[illegible]	[illegible]	2.46105000
.500[illegible]	1[illegible]	2[illegible].00000000
1.[illegible]	50[illegible]	5[illegible].00000000
100.[illegible]	1[illegible]	1[illegible]
451.237[illegible]	[illegible]	1.1[illegible]806510

QUANTITY MAX	VALUE MAX	OBJECTIVE
552.737[illegible]	[illegible]	5[illegible]3.[illegible]

END OF PROBLEM 5

EXAMPLE PROBLEMS UTILIZING PROGRAM TO COMPUTE APPROXIMATE OPTIMUM WEIGHT
AND VOLUME WITH MAXIMUM VALUE

```
                                        INPUT DATA
PROBLEM NO. 1
         191.0            1.0
  6
          24.563         88.88               1.00
         451.237        500.00               2.00
           0.500        100.00               0.01
          88.511        217.83               1.50
           1.000        500.00               0.10
         100.00        1000.00               0.50
PROBLEM NO. 2
         639.9            3.0
  6
          24.563         88.88               1.00
         451.237        500.00               2.00
           0.500        100.00               0.01
          88.511        217.83               1.50
           1.000        500.00               0.10
         100.00        1000.00               0.50
PROBLEM NO. 3
         578.0            2.0
  6
          24.563         88.88               1.00
         451.237        500.00               2.00
           0.500        100.00               0.01
          88.511        217.83               1.50
           1.000        500.00               0.10
         100.00        1000.00               0.50
PROBLEM NO. 4
         552.5            4.0
  6
          24.563         88.88               1.00
         451.237        500.00               2.00
           0.500        100.00               0.01
          88.511        217.83               1.50
           1.000        500.00               0.10
         100.00        1000.00               0.50
PROBLEM NO. 5
         553.0            3.6
  6
          24.563         88.88               1.00
         451.237        500.00               2.00
           0.500        100.00               0.01
          88.511        217.83               1.50
           1.000        500.00               0.10
         100.00        1000.00               0.50
```

SOLUTION

BEGINNING OF PROBLEM 1

FIRST SORT BY DESCENDING INDEX

INDEX	WEIGHT	CUBE	VALUE
20000.0000	.5000	.0100	100.0000
5000.0000	1.0000	.1000	500.0000
20.0000	100.0000	.5000	1000.0000
3.6184	24.5630	1.0000	88.8800
1.6407	88.5110	1.5000	217.8300
.5540	451.2370	2.0000	500.0000

SECOND SORT BY DESCENDING VALUES HAVING IDENTICAL INDEXES

INDEX	WEIGHT	CUBE	VALUE
20000.0000	.5000	.0100	100.0000
5000.0000	1.0000	.1000	500.0000
20.0000	100.0000	.5000	1000.0000
3.6184	24.5630	1.0000	88.8800
1.6407	88.5110	1.5000	217.8300
.5540	451.2370	2.0000	500.0000

LOADING SCHEDULE

WEIGHT	CUBE	VALUE	INDEX
.5000	.0100	100.0000	20000.0000
1.0000	.1000	500.0000	5000.0000
100.0000	.5000	1000.0000	20.0000

MAXIMUM WEIGHT	WEIGHT OBJECTIVE	MAXIMUM CUBE	CUBE OBJECTIVE
101.5000	101.0000	.6100	1.0000

CARGO VALUE = 1600.0000

END OF PROBLEM 1

BEGINNING OF PROBLEM 2

FIRST SORT BY DESCENDING INDEX

INDEX	WEIGHT	CUBE	VALUE
20000.0000	.5000	.0100	100.0000
5000.0000	1.0000	.1000	500.0000
20.0000	100.0000	.5000	1000.0000
3.6184	24.5630	1.0000	88.8800
1.6407	88.5110	1.5000	217.8300
.5540	451.2370	2.0000	500.0000

SECOND SORT BY DESCENDING VALUES HAVING IDENTICAL INDEXES

INDEX	WEIGHT	CUBE	VALUE
20000.0000	.5000	.0100	100.0000

5000.0000	1.0000	.1000	500.0000
20.0000	100.0000	.5000	1000.0000
3.6184	24.5630	1.0000	88.8800
1.6407	88.5110	1.5000	217.8300
.5540	451.2370	2.0000	500.0000

LOADING SCHEDULE

WEIGHT	CUBE	VALUE	INDEX
0.0000	0.0000	0.0000	3.6184
.5000	.0100	100.0000	20000.0000
1.0000	.1000	500.0000	5000.0000
100.0000	.5000	1000.0000	20.0000
88.5110	1.5000	217.8300	1.6407

MAXIMUM WEIGHT	WEIGHT OBJECTIVE	MAXIMUM CUBE	CUBE OBJECTIVE
190.0110	649.9000	2.1100	3.0000

CARGO VALUE = 1817.8300

END OF PROBLEM 2

BEGINNING OF PROBLEM 3

FIRST SORT BY DESCENDING INDEX

INDEX	WEIGHT	CUBE	VALUE
20000.0000	.5000	.0100	100.0000
5000.0000	1.0000	.1000	500.0000
20.0000	100.0000	.5000	1000.0000
3.6184	24.5630	1.0000	88.8800
1.6407	88.5110	1.5000	217.8300
.5540	451.2370	2.0000	500.0000

SECOND SORT BY DESCENDING VALUES HAVING IDENTICAL INDEXES

INDEX	WEIGHT	CUBE	VALUE
20000.0000	.5000	.0100	100.0000
5000.0000	1.0000	.1000	500.0000
20.0000	100.0000	.5000	1000.0000
3.6184	24.5630	1.0000	88.8800
1.6407	88.5110	1.5000	217.8300
.5540	451.2370	2.0000	500.0000

LOADING SCHEDULE

WEIGHT	CUBE	VALUE	INDEX
24.5630	1.0000	88.8800	3.6184
.5000	.0100	100.0000	20000.0000
1.0000	.1000	500.0000	5000.0000
100.0000	.5000	1000.0000	20.0000

MAXIMUM WEIGHT WEIGHT OBJECTIVE MAXIMUM CUBE CUBE OBJECTIVE

126.0635	578.[illegible]	1.61[illegible]	2.0000

CARGO VALUE = 1681.5800

END OF PROBLEM 3

BEGINNING OF PROBLEM 4

FIRST SORT BY DESCENDING INDEX

INDEX	WEIGHT	CUBE	VALUE
20000.0000	.5000	.0100	100.0000
5000.0000	1.0000	.1000	500.0000
20.0000	100.0000	.5000	1000.0000
3.6184	24.5630	1.0000	88.8800
1.6407	88.5110	1.5000	217.8300
.5540	451.2370	2.0000	500.0000

SECOND SORT BY DESCENDING VALUES HAVING IDENTICAL INDEXES

INDEX	WEIGHT	CUBE	VALUE
20000.0000	.5000	.0100	100.0000
5000.0000	1.0000	.1000	500.0000
20.0000	100.0000	.5000	1000.0000
3.6184	24.5630	1.0000	88.8800
1.6407	88.5110	1.5000	217.8300
.5540	451.2370	2.0000	500.0000

LOADING SCHEDULE

WEIGHT	CUBE	VALUE	INDEX
0.0000	.0000	0.0000	20000.0000
0.0000	.0000	0.0000	3.6184
0.0000	.0000	0.0000	1.6407
1.0000	.1000	500.0000	5000.0000
100.0000	.5000	1000.0000	20.0000
451.2370	2.0000	500.0000	.5540

MAXIMUM WEIGHT	WEIGHT UTILIZATION	MAXIMUM CUBE	CUBE OBJECTIVE
552.2370	552.[illegible]	2.6000	4.0000

CARGO VALUE = 2000.0000

END OF PROBLEM 4

BEGINNING OF PROBLEM 5

FIRST SORT BY DESCENDING INDEX

INDEX	WEIGHT	CUBE	VALUE
20000.0000	.5000	.0100	100.0000
5000.0000	1.0000	.1000	500.0000
20.0000	100.0000	.5000	1000.0000
3.6184	24.5630	1.0000	88.8800
1.6407	88.5110	1.5000	217.8300

.5540	451.2370	2.0000	500.[illegible]

SECOND SORT BY DESCENDING VALUES HAVING IDENTICAL INDEXES

INDEX	WEIGHT	CUBE	VALUE
20000.0000	.5[illegible]	.010[illegible]	100.[illegible]
5000.0000	1.0000	.1[illegible]	500.[illegible]
20.0000	100.[illegible]	.5[illegible]	1000.[illegible]
3.6184	24.56[illegible]	1.[illegible]	[illegible].8800
1.6407	[illegible]8.5110	1.9000	217.[illegible]
.5540	451.[illegible]	2.0000	500.[illegible]

LOADING SCHEDULE

WEIGHT	CUBE	VALUE	INDEX
0.0000	.[illegible]	.0000	3.6184
0.0000	.[illegible]	.0000	1.6407
.5000	.[illegible]	[illegible]	20000.[illegible]
1.0000	.1000	500.0000	5000.[illegible]
100.0000	.5000	1000.0000	20.[illegible]
451.2370	2.0000	500.0000	.5540

MAXIMUM WEIGHT	WEIGHT OBJECTIVE	MAXIMUM CUBE	CUBE OBJECTIVE
553.7370	553.[illegible]	2.6000	3.6[illegible]

CARGO VALUE = 2[illegible].0000

END OF PROBLEM 5

SEMI-LOGARITHMIC PROBLEM

INPUT DATA

```
00170009
00010001-1.0
00010002-0.2310483333333
00020001 1.0
00020002 0.462097777777778
00020003 1.0
00030001 1.0
00030002 0.3837650
00030004 1.0
00040001 1.0
00040002 0.17270857
00040005 1.0
00050001 1.0
00050002 0.229920
00050006 1.0
00060001 1.0
00060007 1.0
00070001 1.0
00070002 0.3583720
00070008 1.0

00020003 2.07944
00030004 2.30259
00040005 1.20896
00050006 1.60944
00060007 0.69315
00070008 1.79176
```

SEMI-LOGARITHMIC PROBLEM

SOLUTION

```
FUNCTIONAL            1.38444540
VARIABLE            VALUE
   7               .00369576
   4               .45877424
   2              3.00799060
   6               .22838856
   1               .68945424
   8               .02432611
VARIABLE         SHAD. COST
   3               .20159620
   5               .79840390
```

LOGARITHMIC PROBLEM

INPUT DATA

```
0007000 9
00010001-1.0
00010002-0.58896[illegible]421
00020011.0
00020021.7[illegible]945885
00020031.0
00030011.0
00030021.113309307
00030041.0
00040011.0
00040021.0
00040051.0
00050011.0
00050022.6[illegible]056[illegible]14
00050061.0
00060011.0
00060071.0
00070011.0
00070021.47374[illegible]84
00070081.0

00020032.30259
00030041.79176
00040051.94591
00050061.38629
00060070.91629
00070082.07944
```

LOGARITHMIC PROBLEM

SOLUTION

```
FUNCTIONAL            1.46[illegible]
VARIABLE              VALUE
    7              .00644002
    4              .00188834
    5              .0447061[illegible]
    2              .791353[illegible]
    1              .009249[illegible]
    8              .0033359[illegible]
VARIABLE          SHAD. COST
    3              .08369766
    6              .91630240
```